A Smart Rogaining Experience for Orientation to Computer Science

Manuela Chessa [1], Giorgio Delzanno [1], Angelo Ferrando [1], Luca Gelati [2], Giovanna Guerrini [1], Viviana Mascardi [1], Nicoletta Noceti [1], Francesca Odone [1], and Francesca Vitali [3]

[1] *DIBRIS, University of Genoa, Italy*
[2] *Edutainment Formula, Italy*
[3] *DNBM, University of Verona, Italy*

Correspondence*:
Giorgio Delzanno
giorgio.delzanno@unige.it

ABSTRACT

In this paper we address the problem of designing new formats of computer science orientation activities to be offered during high school students internships in our bachelor degree. In order to cover a wide range of computer science topics as well to deal with soft skills and gender gaps issues, we propose a teamwork format, called smart rogaining, that offer engaging introductory activities to prospective students in a series of checkpoints dislocated along the different stages of a rogaine. The format is supported by smart mobile and web applications. Our proposal is aimed at stimulating the interest of participants in different areas of computer science and at improving digital and soft skills of participants and, as a side effect, of staff members (instructors and university students). In the paper we introduce the proposed format and discuss our experience in the editions organized at the University of Genoa before the pandemics (2019 and 2020).

Keywords: Computer Science, Computer Science Education, Orientation, Recruitment and Retention, Technology Enhanced Learning, Teamwork, Computational Thinking

1 INTRODUCTION

1.1 Background and Motivations

Computer science specialists are needed all over the world. Despite the wide range of available professional figures, the number of graduates does not match the demand of the labor market [26]. Furthermore, education systems are dealing with the impact of technology and the fast changing need for new skills and knowledge. In the last ten years, the problem of preparing young people for careers in computer science has thus attracted considerable attention. Many important initiatives have been proposed by non profit international organizations, some of which gained worldwide popularity such as the

the Code.org initiative [43, 44, 10] and the joint Informatics Europe and ACM Europe Working Group on Informatics Education [6, 7, 19]. In April 2022, the project Programma il Futuro [38] inspired by Code.org and by the Computational Thinking principles [55] received the first prize for promoting digital competences in Italy.

To support computer science education, a wide range of new technologies and applications have been developed both in industry and academy. For instance, the Scratch Foundation guided by Mitchel Resnick, Professor of Learning Research at the MIT Media Lab, in collaboration with Google developed Scratch [49] a very popular visual programming languages for beginners [34, 46]. The Catrobat organization headed by Professor Wolfgang Slany worked for more than ten years to a mobile version of the Scratch language called PocketCode [37, 8]. The Micro:bit Foundation designed a series of programmable and wearable devices for beginners [50].

More in general, initiatives for promoting computer science are nowadays organized by almost all schools and academic institutes around the world [15]. Formats such as scavenger hunt [54, 25, 35], role games [5], online polls [22] are becoming more and more popular especially when combined with the use of emerging technology (mobile apps, wearable sensors, virtual reality, etc).

1.2 Research Goal

Our overall goal is to increase the number of computer science students in our University, and to enroll informed students, since our degree (as most Computer Science Bachelor's degrees in Italy) is characterized by a high dropout rate. In this respect internships for high school students play a crucial role: they represent the last chance to capture the attention of undecided prospective students, and to provide an exposure to the topics that will be studied in the degree. Internships have a limited duration (less than one week) and combine structured activities (seminars, meetings with instructors, hands-on lab, mini-projects, etc) with half-day slots in which it is possible to experiment new formats.

Our specific research goal is to design reproducible formats for half-day activities that could help prospective students in appreciating the beauty, technical contents, and professional potential of our discipline. All activities shall be organized by academic teachers in collaboration with University students and they shall be offered to high school students with different background, and potentially completely novices to computer science.

Despite the large number of existing computer science dissemination initiatives, our research question has to face some important issues:

- *Topic Coverage*: A first challenge is to provide, in a short time activity, a sufficiently broad overview of the different topics covered in a computer science degree, with a right balance between methodological and technological aspects. To appreciate the difficulty, the 2012 Computing Classification of the Association for Computing Machinery[1] gives a comprehensive list of subject categories of our discipline. They range from the Theory and Mathematics of Computation to Applied Computing (education, law, economy, healthcare, etc.) and Social and Professional Topics.

[1] https://dl.acm.org/ccs

- *Soft-skill*: A second challenge is to convince prospective students that our discipline also requires soft skills, e.g., teamwork and good interaction skills [32], combined with the ability to adapt to the rapid evolution of technology. In addition, as an orientation activity for university studies, we aim at stressing the importance of soft skills that are crucial for higher education, such as planning, time management, goal prioritization, work under stress, reaction to unexpected events.
- *Gender-gap*: A closely related challenge is finding strategies for the mitigation of gender gap, a problem for the vast majority of computer science degrees [23].

1.3 Contribution

To face the above mentioned challenges (topic coverage, soft-skills, and gender-gap), in this paper we present a novel teamwork format for internships of limited duration (half day) inspired by rogaining outdoor activities and, more in general, by strategic team reasoning. Gamification drives the design and implementation of our approach: participants, divided into teams, are involved in a rogaine consisting of several checkpoints organized as a series of practical activities of the same duration. The activities proposed in the checkpoints cover different subject categories of our discipline and combine soft-skills, technology, and computational thinking tasks. Differently from scavenger hunt, rogaining requires some form of strategic reasoning since teams are forced to select a strict subset of checkpoints to visit during the competition. Indeed, by design, the total duration of all activities is larger than the duration of the rogaine.

Our format is aimed at improving student engagement with respect to traditional internships activities. Rogaining [45] is an outdoor orienteering sport involving both route planning and navigation between checkpoints using a variety of map types. In a rogaine, teams usually consisting of two to five members choose which checkpoints to visit within a time limit with the intent of maximising their score. Although endurance and competition are important aspects, teamwork is probably the central feature of this sport. Rogaining is indeed one of the favourite activities in corporate events organised by companies. Our format is mainly designed as an indoor navigational activity with a limited time duration (4 hours). During the event, students need to complete a number of checkpoints, consisting of both plugged and unplugged computer science activities. Students are exposed to computer science both with learning goals (through labs and hands-on activities on basic coding concepts) and dissemination goals, to let them approach more advanced and challenging topics through short seminars, demos, and exhibits. Since 2019 and before the Covid-19 pandemics, the format has been applied in orientation events for high school students and freshmen of the Computer Science bachelor degree of our University.

The proposed format is the result of a joint design effort between Computer Science instructors, team-building experts (Edutainment Formula) and psychologists. Teamwork turned out to be quite effective for stimulating student engagement in hands-on activities and projects, especially considering that our interns are part of a large heterogeneous group of students. More in general, the format lets the participants exercise different soft skills, ranging from intra-personal skills like adaptability and flexibility, to inter-personal skills like negotiation, to methodological skills like problem solving.

1.4 Smart Technology

The management of the activity is supported by a mobile web application, namely the SR-App (Smart Rogaining App), developed in collaboration with Edutainment Formula. Its functionalities have been tested following the orchestrated crowdsourced testing approach described in [31]. SR-App allows teams to explore a map with information on checkpoints (description, scores, etc), reserve for a single checkpoint activity, and check whether a checkpoint is currently locked by another team or not.

1.5 Experimental Validation

In this paper, starting from a detailed description of the proposed teamwork orientation format, we will discuss soft skills goals for participants and mentors, orienteering goals, computer science education goals, and the benefits of introducing a technology support both in practice (on the field) and in theory (in the design phase). Furthermore, we elaborate on data collected for an experimental evaluation in in the 2019 and 2020 editions at the University of Genoa with 135 and 62 participants, respectively. We believe that a discussion of our experience could be helpful to evaluate pros and cons for reproducing a similar format in other institutes and, perhaps, in other disciplines. Since bachelor, master and PhD students are involved as active part of the organization and as mentors for the checkpoint activities, our format also offers the opportunity of improving soft skills of university students. The paper presents in an integrated and extended way preliminary works mainly discussed in a workshop [13, 14, 40, 3].

1.6 Plan of the Paper

The remainder of the paper is organized as follows.

- In Section 2 we introduce our teamwork rogaining format as well as learning goals for both participants and university students. We also discuss the advantages in using the support of a webapp in conducting the event.
- In Section 3 we describe the challenges used in the two editions.
- In Section 4 we discuss the experimental evaluation of the format based on relevant data collected in the selected editions.
- In Section 5 we discuss and compare our approach with related work.
- Finally, in Section 6 we address conclusions and future directions for our research.

2 SMART ROGAINING FOR ORIENTATION TO COMPUTER SCIENCE

In this section, we present our format and discuss the expected learning outcomes for participants and staff members (University instructors and students) and the relevance of having the activity supported by a web app, that makes the rogaining *smart*.

2.1 Rogaining Format

Our teamwork format is based on a rogaining activity with a duration of 4 hours. The rogaine consists of a collection of checkpoints distributed in different areas of our campus.

In the two reported editions we selected rooms from different buildings, floors, departments in the campus of the Science Faculty of our University in order to get closer to traditional navigational competitions. A score is assigned to each checkpoint based on the distance from the home base and on the difficulty of the proposed exercise. As we will describe later in this section, in our experience with high school interns we proposed different types of challenges ranging from problem solving to tinkering, coding and programming labs. In general, it would be desirable to assign different roles or tasks to group members in each challenge. Prior to starting with the activity, participants are required to fill in a questionnaire to identify their background, skills, and aspects of individual personality that are used to form the teams. The goal is to group students in homogeneously heterogeneous teams with respect to different axes, e.g., different schools, background, skills, so as to balance the competition during the game. Specific attention is paid to gender balancing and to avoiding putting interns from the same school or already knowing each other in the same team.

During an initial briefing, a map of the event location and a short description of the checkpoint catalogue is assigned to each team. The number of checkpoints must be greater than the number of teams (at least 20%). Furthermore, teams are forced to select a subset of checkpoints, i.e., the total duration of the event must be much less than the total time required by all checkpoints. In the two editions we proposed 15 checkpoints for 12 teams and 12 checkpoints for 9 teams, respectively. The duration of the activity at each checkpoint is 30 minutes, so the target number of checkpoint to complete is 6-7. Teams have 20 minutes to select an initial strategy, i.e., to select the checkpoint list to visit during the challenge. Before starting the game each team has to communicate to the game staff a final score prediction based on their strategy. The strategy itself is kept secret. The score prediction will be useful in the post-game briefing. Two mentors are assigned to each checkpoint. They are in charge of the following tasks: explain the activity and the exercise, assign the points to the team, evaluate the behaviour of team members, and, finally, notify that the lock has been released to all other teams. Staff members have to provide support for implementing this kind of non trivial interaction between different teams and checkpoints.

Checkpoints are locked by teams via the SR-App. Teams are required to lock the next checkpoint right after the conclusion of the current one. Therefore, teams may have to dynamically modify their strategy and navigation plan.

When the rogaine time expires, teams return to the home base for the conclusion of the game. The organization staff collects score and timings of each group and presents the final ranking of the game comparing expected and achieved results. Participants are then required to fill in a peer observation form (in the spirit of Bales Interaction Process Analysis [2]) for each member of their team.

An UML-like diagram of the workflow associated to the proposed activity is shown in Fig. 6 in Appendix 1 together with additional specifications of roles and role tasks.

2.2 Learning Goals for Teams and University Students

Checkpoint activities are proposed by volunteering bachelor, master and PhD students in collaboration with staff members. This feature has the nice effect of introducing a novel type of soft skills activities in our degrees. Indeed, in most of the cases the design of a

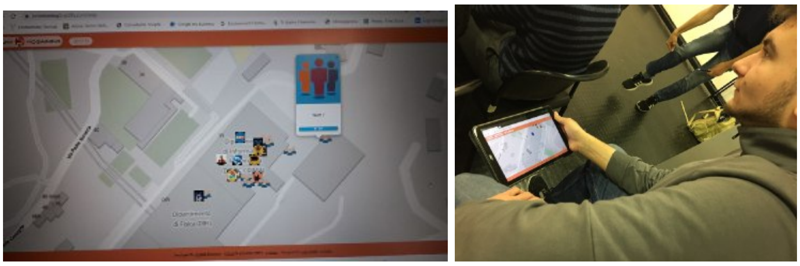

Figure 1. SR-App: Campus map with challenges locations (left), and a team leader with the SR-App (right).

checkpoint lab requires a software artefacts specifically designed for computer science education. This task turns out to be a quite non standard, but still interesting, exercise for our students. Mentors have to deal with organisational issues in order to manage teams and have to strictly interact with staff members before, during and after the game. This creates new communication channels between students and staff members that go beyond what is required in a standard study program (teamwork in teamwork!). In both editions, mentors at checkpoints running the activities are students. Mentors are required to evaluate, via simple questionnaires, the level of engagement and the degree of success in completing the activity of each team, thus providing feedback to the entire organisation process.

The game structure itself embeds non trivial computer science concepts. For instance, rogaining is built on top of an important concurrent programming pattern, i.e., synchronization. Finding the initial plan is an instance of a variation of the well known traveling salesman problem with additional constraints induced by team members background and skills.

During the game at any given time, each checkpoint is assigned to a unique team. Since it is not reasonable to lock all checkpoints in advance, teams are required to lock the next checkpoint right after the conclusion of the current one. Race conditions are admitted in this phase. They are solved by using a FIFO discipline with no pending queues, i.e., if the planned checkpoint is locked, teams have to select another free checkpoint and retry in the next round.

Teams have no tutors during the game. Instead they had to follow instructions, maps and signs, understand the rules, ask questions to other teams and mentors. The event is an anticipation of their future student life, in which they will be required to move from one lesson to another, schedule their time for labs and exams, etc. Most important, they experience that sometimes is not possible to complete everything in due time, that they may need to give up something and take decisions, and that plans need to be realistic. Carefully planning is crucial, but unexpected events may happen (e.g., the checkpoint we wanted to book is not available) so teams have to dynamically modify their navigation plan, a frequent situation in a student career.

Technology enables a diversification of activities, different perspectives for common concepts (e.g. coding via visual languages, modal interfaces, etc), animate the checkpoint activities, provide interesting links for possible insights on theoretical aspects. We decided to introduce the rogaining model only after the adoption of a smart app (SR-App) to manage the entire event. The app is used by teams and mentors and provide functionalities to lock (teams) and unlock (mentors) checkpoints, and to notify these operations and the current score of each team to all participants. All these information are shared among all participants via a map embedded in the user interface (see Fig. 1). The current game state is displayed on a large screen in the home base so that game manager can monitor both teams and checkpoints. Telegram is used for fast communication among staff members. Each team is equipped with at least one tablet used to monitors the entire game using the SR-App as shown in Fig. 1. The app is used for navigational purposes (to find the checkpoints), to lock and release a checkpoint.

The introduction of new game features was enabled by the supporting SR-App. Specifically, the possibility of dynamically assigning checkpoints to teams (i.e., locking/unlocking of checkpoint etc) is an easy task for a centralized management of the event via the SR-App, while it could be a very hard task to implement via other communication tools such as Whatsapp and Telegram without resorting to bot or similar artefacts. On another level, the fact that the web app itself had been developed by our students, was helpful to demonstrate to interns what a computer science student can realize in a Bachelor final project.

3 CHECKPOINT ACTIVITIES

To give a better idea of the variety of activities that can be included in our format, in this section we briefly discuss the challenges organized by staff members and computer science students in the two reported editions (2019 and 2020 editions during a one week internship for high school students at the University of Genova). Specifically, we first summarize the entire set of challenges and then focus on and discuss in greater detail three of them, designed by our students.

The activities offered in the checkpoints give an overview of foundational and applied aspects of computer science. Technology is used as a vehicle to show the importance of algorithms and programming concepts, they are the engine behind most of the technology we used today. Checkpoints combine computational thinking aspects with mathematics, physics, and engineering concepts. At the same time, they show different application domains such as simulation and serious games, automation and IoT, data analysis, smart applications, etc. A summary of the proposed challenges is described in Table 1, while Table 2 describes the topic and knowledge elements of each activity, according to the ACM/IEEE CC 2020: Computing Curricula 2020[2], and reports the internship edition in which the activity was proposed.

[2] Computing Curricula 2020, CC Task Force https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf

Name	Description
Vudù	Inspired by the board game with the same name, teams are challenged with questions on logic and computer science principles.
Color-Run	Participants have to find an algorithm for solving the map coloring problem, exponential complexity, with the minimum possible number of colors.
Pachinko	Inspired by Galton's Machine, the goal is to write a Scratch program to visualize the Gaussian probability distribution.
The-Mind	Aimed at stimulating team work in collaborative problem solving tasks, it is inspired to the card game with the same name.
Memory	A cognitive task in which participants challenge each other in order to store the maximum number of piece of data (colors, words, numbers, etc) in the short-term memory.
Calcolemus	Participants got involved in programming tasks using the Sketchware App[3] that provides a visual language for creating Android mobile applications.
Pinball-Wizard	Teams had to customize a flipper simulator in the Pocketcode[4] app so as to make it controllable using tablet gyroscope and accelerometer.
Heartbeat	The goal is to create a pedometer prototype using the wearable microcontroller Microbit[5] that comes equipped with an accelerometer. An initial template in the makecode visual language[6] is provided to participants.
Fast-and-Furious	Teams have to modify the controller of an Arduino car in order to boost its engine when required by the user via a predefined smartphone app.
Micropython	Participants have to solve programming tasks related to explore the functionalities of the Micro:Bit microcontroller including radio communication via the Micropython library[7]).
Dashboarding-is-not-a-crime	Teams have to use the Node-red graphical environment[8] and the educational version of the Ubidots IoT platform[9] to build a web dashboard to visualize aggregated analysis of data acquired from sensors.
Catch-the-flag	A problem solving challenge inspired by cybersecurity: Participants have to decrypt secret messages exchanged between airplane pilots and control tower operators with the help of a series of hidden clues.
3D-Coding	Teams are required to create a model of a 3D logo using Beetleblocks[10], a coding tool for 3D drawings. The model is then used to print the logo in 3D.
Hacking-Videogames	Teams are required to create, using Scratch or Python, a custom bot in the RLBot platform[11] to modify the behavior of a racer car in the Rocket League videogame[12].
Skeleton	Teams are required to apply the DeepLabCut [13] engine, a deep learning architecture, to identify parts of a body in video frames so as to perform simple analysis of human movement such as walking.
Tangible-Coding	Participants use a collection of physical shapes as a real programming language. Tangible programmers are given a set of simple and colored 3D shapes representing nouns and verbs, and very simple rules to build the tangible sentence (i.e. the program). Participants learn basic concept such as sequence, the effect of changing the order of elements, and the power of selection constructs.
School-of-Rock	Participants create music instruments for their rock band using Makey Makey[14] and Scratch[15].
Whiplash	Teams create sounds and rhythms using the concurrent language Sonic Pi that provides Ruby programming libraries for live coding[16].
Codinji	Inspired to the Jumanji movie, participants use Scratch 3.0 to create a videogame based on simple blocks for webcam motion capture to insert the video captured by the webcam as a background for a Scratch game.
Alien-vs-Terminator	The challenge was based on a VR game specifically designed by computer science students in Unity and implemented for collaborative teamwork.

Table 1. Checkpoint Description.

3.1 Checkpoints Designed by University Students

In this subsection, we provide some additional details on three checkpoints that were specifically designed for their use in the smart rogaining competition by University students as projects of bachelor/master courses. The examples show a concrete example of learning

Name	Topic	Knowledge	Editions
Vudù	Logic and Computer Science	ACT	2019
Color-Run	Algorithms and Complexity	ACT, PSTS	2019,2020
Pachinko	Algorithms and Complexity	ACT, MS	2019,2020
The-Mind	Intelligent Systems	CT, MS	2019
Memory	Intelligent Systems	ACT, MS	2019
Calcolemus	Software Development	PSTS	2019
Pinball-Wizard	Software Development	ACT, CT	2019,2020
Heartbeat	Embedded Systems	ACT, CT	2019
Fast-and-Furious	Embedded Systems	ACT, CT	2019
Micropython	Embedded Systems	ACT, CT	2020
Dashboarding-is-not-a-crime	Internet of Things	ACT, CT	2019,2020
Catch-the-flag	Security Issues and Practice	CT, EIP, PSTS	2020
3D-Coding	Computer Graphics	CT, PSTS	2019
Hacking-Videogames	Computer Graphics	CT, PSTS	2020
Skeleton	Computer Vision/AI	CT, EIP	2019,2020
Tangible-Coding	Computer Vision/AI	ACT, CT	2019,2020
School-of-Rock	Tinkering	CT, ACT	2019,2020
Whiplash	HCI/Programming	CT, PSTS	2019
Codinji	HCI/Programming	CT, PSTS	2019,2020
Alien-vs-Terminator	Virtual Reality	CT, PSTS	2019,2020

Table 2. Knowledge elements from the ACM/IEEE Computing Curricula 2020: ACT=Analytical and Critical Thinking; EIP=Ethical and Intercultural Perspectives; CT=Collaboration and Teamwork; MS=Mathematics and Statistics; PSTS=Problem Solving and Trouble Shooting.

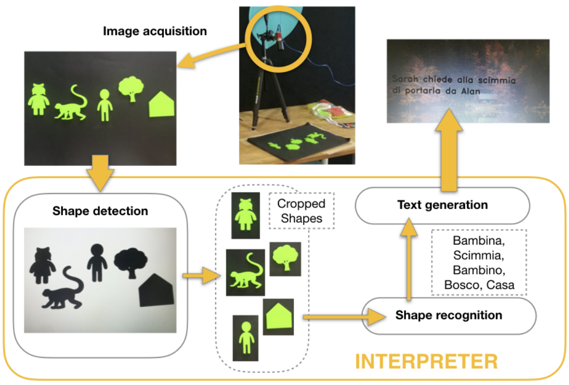

Figure 2. The main steps of our software components, on an example (bambina=girl, bambino=boy, scimmia=monkey, bosco=woods, casa=house).

outcomes for computer science university students: designing application that can be tested on the field with large number of users.

The first example is Tangible Coding, a programming activity based on physical shapes used as instructions [18, 24, 48]. On the back-end of the tangible coding activity, there

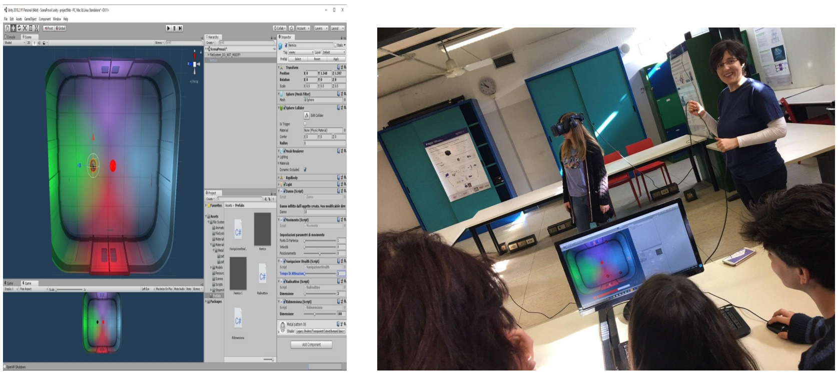

Figure 3. (a) The Unity interface presented to the attackers' team: top view of the game and interface to create custom enemies. (b) The VR player tries to dodge the enemies created by the group in the foreground (the attackers).

was an interpreter of the objects sequences composed by different artificial intelligence modules (see Fig. 2, right), going from the acquisition of an image depicting the sequence of objects, to the localization and recognition of the shapes on the image, to the generation of a fantasy short story using Natural Language Processing principles. The shape recognition task has been addressed with a Convolutional Neural Network[17] we designed and trained from scratch, collecting a set of shape images under different environmental conditions (∼7300 images, the 80% of which has been used for training, the remaining samples for validation). We also included a data augmentation procedure to increase the robustness against shape orientations variability. The model has been assessed directly on the field during lab sessions, achieving the 89% of recognition rate. The shape recognition returns an ordered list of labels which is fed to the following text generation module to produce a sentence in a natural language (italian), following a simple rule based approach. Participants were presented the technical details of the back-end technology, and then they were asked to test the system.

The second example is the Codinji challenge. The webcam blocks available in Scratch 3.0 are used to detect video motion in the current position of a sprite in order to simulate the interaction between the real player and the sprites in the game. This feature can be used to create games involving groups of participants. The Scratch application with the webcam video as a background was projected in a large screen in front of participants filmed by a webcam. Two teams can then play against each other in order to catch different types of sprites floating in the game board.

Finally, in the Alien vs Terminator challenge, one student (the VR player, see Fig. 3(b)) wears an HMD [18] for VR and is immersed in a virtual environment, where some sphere-shaped enemies attack him from every direction. The remaining participants are split into two teams: defenders and attackers. Defenders monitored the virtual scene from a

[17] https://keras.io/

[18] HTC Vive Pro headset

top viewpoint, and their mission was to help the person immersed in VR to dodge the enemies. Attackers' objective was to outsmart the defenders' team communication and hit the VR player by creating custom enemies with special features, e.g., different mesh and collider size, velocity, or transparency of the texture, using the Unity editor. Specifically, the attackers' team had to use the Unity editor in play mode (the debug tool built in the editor is shown in Fig. 3(a)) as if they were real developers. We created a reusable asset (prefab) of a standard enemy, which the students could instantiate at run time, and several scripts. By default, the enemy prefab was disabled in the scene. The students had to instantiate an enemy, modify its parameters and behaviors, and finally enable it to finalize the spawning process. The scripts we provided modified e.g. mesh size, collider size, speed, spawning position, and direction of movement. In the beginning, a tutor briefly explains the basics of the Unity editor interface: how to assemble custom enemies using pre-built scripts, tweak their parameters and activate the created game objects (~5 minutes). After the game starts, the attackers have about 10 minutes to defeat the VR player. The defenders win if the VR player survives. After one match, the two teams switch roles, and a new VR player is chosen.

4 EXPERIMENTAL EVALUATION

In this section we discuss the evaluation of the proposed model in terms of data collected from the 2019 and 2020 internships editions. After shortly describing the setup, i.e., providing some details about the editions, we discuss the results in terms of teamwork evaluation, rogaine evaluation, effectiveness for recruitment and orientation and effectiveness for ensuring informed enrollment and contrasting dropout.

4.1 Experimental Setup: Analysed Editions

Interns are high school students enrolled in 12-th grade (majority) and 13-th grade. The period was early February. Table 3 summarizes basic facts about the editions.

Year	**2019**	**2020**
students	135	62
females	44	18
12th grade	80	17
distinct schools	37	28
teams	12	9
checkpoints	15	12

Table 3. Basic facts about the two reported editions

4.2 Teamwork Evaluation

Teams were asked to declare their scores before starting the rogaine. Table 4 report declared (before game start) and obtained scores for the teams in the two editions. Note that all the teams quite relevantly underestimated their performances. This is quite typical in this kind of activities. Teams were overall quite cautious in predicting their scores. This could be due to the fact that teams are formed by students that did not know each other before starting the activity.

Edition	Average decl	Stdev decl	Average obt	Stdev s obt	Delta (obt-decl)	Ratio (obt/decl)
2019	580	57,74	803	42,79	+ 223	138,4%
2020	596	121,29	936	93,83	+ 340	157%

Table 4. Declared (decl) and obtained (obt) scores for teams in the two editions.

At the end of the activities, a peer evaluation is performed [17, 41]. Figure 4 summarizes the results of the peer evaluation questionnaire aggregating student responses according to the three relevant dimensions: climate, productivity and process. The final questionnaire contained eighteen questions, six per dimension. Students are required to observe teams with respect to different behaviors. Possible answers are: behavior observed, opposite behavior observed, no observation for this behavior. Examples of questions are "Encourage, support and help others" (climate), "Quickly acquire information, learn from others" (productivity), "Work out analysis and evaluation on costs/benefits on multiple possibilities" (process). Note that an ideal team should have a perfect balance (33%) of the three dimensions. The fact we got a light unbalance towards climate is coherent both with the age of the participants and with the joyful atmosphere during the activity.

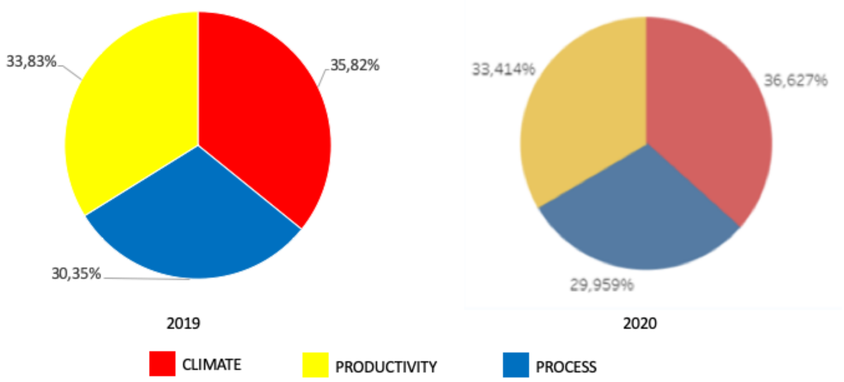

Figure 4. Dimensions in teams, according to the peer evaluation questionnaire, in the two editions

We also analysed the evaluation that mentors assigned to each team during the entire activity. The mean scores turned out be: 4.5 for engagement and 4.7 for the activity completion (in both cases in a 1-5 scale). As a positive outcome we observed a very low variance among different teams and different checkpoint: all computed average values by team and by checkpoints are comprised between 4 and 5. This outcome seems to validate the criteria that we adopted for team formation and for the selection of the checkpoint activities.

4.3 Rogaine Evaluation and Student Appreciation

Figure 5 reports the appreciation for the activity by the interns, obtained from an anonymous post-internship questionnaire. The figure reports the overall evaluation for the rogaining activity, the average evaluation for all the other activities proposed during the one-week internship, and average, minimum, and maximum evaluation of the individual checkpoints. All evaluations are on a scale 1 (I didn't like it at all) to 5 (I liked it a lot).

We also collected data during the rogaining contest via the SR-App. More precisely, the sequence of visited checkpoints for each team and the time required to complete each task. These data turned out to be quite useful in order to integrate the parameters selected via the agent-based simulation of the rogaine with a finer tuning of duration and physical location of each checkpoint. We also exploit the large number and high frequency of the requests to the SR-App as a stress test for the SR-App itself. Logged data related to use of the SR-App user interface have been employed in order to test and improve the usability of the SR-App itself using the crowd-sourced testing methodology explained in [31]. All the internship

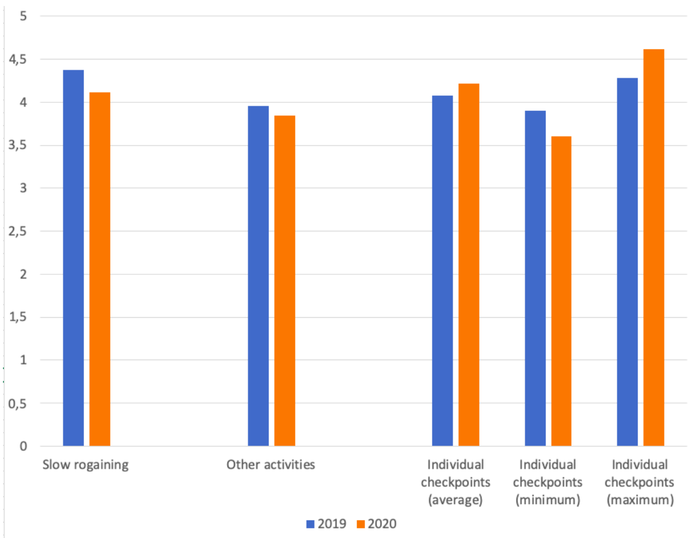

Figure 5. Student appreciation

activities were very well received, but the rogaining was one of the most appreciated ones (in the top-2 for both the editions). All checkpoints received very positive evaluations.

4.4 Rogaine Effectiveness for Recruitment and Orientation

The main motivation for organizing such activities is ensuring informed enrollment and contrasting dropout. We thus measure the effectiveness of the proposed events in terms of (i) percentage of the participants that changed their mind in terms of enrolling/not enrolling in the Computer Science BSc after participation to the internship (collected through a pre/post questionnaire) (ii) analysis of the careers of students that participated to the events and then actually enrolled to our degree.

Before presenting the data, we need to point out two limitations. First of all, as already discussed, the rogaine events are part of a longer internship and what we are discussing here is the effectiveness of the entire internship, not of the rogaine alone. Second, in order to attract students from different geographical area, the team participants are from different areas. After analysing our data, we realize that almost only students from close areas actually enrolled to our university. We are able to monitor the career only of local students, thus the analysis of careers is partial (and on a limited sample).

In terms of effectiveness for orientation and recruitment:

- 39.4% of the participants that then actually enrolled to our degree declared an effect of the internship on the decision of enrolling in the Computer Science BSc (from mildly interested to strongly interested),
- 15.15% of the participants that then actually enrolled to our degree declared a strong effect of the internship on the decision of enrolling in the Computer Science BSc (from not interested to strongly interested).

Information are collected by internship pre and post interviews and we restrict to participants then are currently actually enrolled in our degree. Note that there is also a positive orientation effect of the internship in terms of discouraging enrollment of participants with a misconception of programming and computer science.

In terms of career analysis, some results are reported in Table 5. The table report:

- the number of participants to one of the editions of the internship that enrolled to one of the possible cohorts, and those that are still enrolled[19];
- the dropout rate, which is lower than the dropout rate of the entire population, which is around 23%;
- the female numbers and percentages of initial and current enrollments (higher than the percentage on the overall enrollments, which is 10-12%);
- the average percentage of acquired ECTS over the total ECTS that can be acquired (much higher than the average on the entire enrolled students, which is 27,8%).

The lower dropout rate for 2021 can also be due to the fact that we are observing just one semester, while students of cohort 2019 are now in the 6th and last semester of the Bachelor.

[19] Since our internship involved students in 12th and 13th grade, 2020 cohort collected students from both the editions, while 2019 cohort only students that were attending 13th grade in 2019 edition, and 2021 cohort only students that were attending 12th grade in 2020 edition.

The lower percentage of acquired ECTS for cohort 2020 compared to 2019 and 2021 can be likely due to the effect of distance learning and COVID-19 pandemic, a similar effect can be observed in the entire student population. Overall, the much higher percentages of female students and of acquired ECTS seem to give further strength to orientations initiatives aimed at increasing both engagement and coverage of the different areas of computer science.

cohort	enrollments		dropout rate	female enrollments		avg ECTS
	freshmen	current		freshmen	current	
2019	13	11	19%	2 (15%)	2 (18%)	85%
2020	28	23	18%	5 (18%)	5 (22%)	61%
2021	10	9	10%	3 (30%)	3 (33%)	81%

Table 5. Enrollments and careers of internship participants. The total number of enrolled students per year is as follows: 232 in 2019, 241 in 2020, and 286 in 2021.

5 RELATED WORK

This section discusses most relevant related work. It starts discussing general studies on Computer Science orientation, including gender issues. We then focus on gamified and app-supported activities, most notably scavenger hunts, that are close to the format we propose. We then specifically discuss work related to two specific challenges of the proposed activity, namely topic coverage (including elements of novelty of the checkpoint activities designed by our students) and soft-skills.

5.1 Orientation to Computer Science and Gender Issues

Recruiting and retaining STEAM majors is an ongoing challenge for colleges and universities. The issue is even more relevant for Computer Science and IT higher education, given the shortage of qualified workforce [12, 47] and [36] for a report in the Italian education system.

The issue of how to attract students have been investigated, with specific attention to gender issues [9, 33], attributing the reasons of gender unbalance mainly to insufficient early experience, lack of role model, and stereotypes. More in general, gender issues in STEM are attributed to the importance of communal or other-oriented goals for female students. In this respect, our format fights stereotypes by putting emphasis on teamwork and communication, and promotes self-efficacy by the solution of simple assignments associated with checkpoints and is well suited to covering diverse topics related to other-oriented domains (like, e.g., environmental and sanitary applications). Offering an internship or participation to a summer camp earlier in a student's undergraduate career [1] not only allows participants to gain confidence in their ability to apply their skills to real world problems, but have been demonstrated useful in piquing interest, exposing students to different topics.

The longer the activity, the highest the effectiveness, our challenge is to design an activity that does not require too much time (e.g., [28] that lasts a whole semester).

A relevant related problem is the high dropout rate in IT higher education studies [27, 26, 21]. Among the most frequently proposed solution to mitigate this problem we found tutoring and early assistantship programs.

5.2 Gamification, Rogaining, and Scavenger Hunts

The advantages of gamification in education and in computer science education specifically, have been thoroughly investigated [20, 30, 4]. For instance, gamification has been frequently applied in order to improve student retention [54]. In this context Augmented/Virtual Reality (AR/VR) and Internet of Things (IoT) technology can substantially improve the student experience especially when the activities are organized in an University environment [25, 35]. Game-development approaches have been proposed also in introducing novices to coding [42], bringing advantages in terms of motivation, fun, commitment, and enthusiasm. Gamification is being used also for orientation and formats such as scavenger hunt [54, 25, 35] are very common for university orientation, but also role games [5] and are more and more frequently combined with the use of emerging technology (mobile apps, wearable sensors, virtual reality, etc) [16, 22].

A unique feature of our format is the association of computer-related hands on activities to checkpounts, thus combining elements coming from early internships, assistantship programs, and gamification principles applied in the recruitment phase (orientation activities for senior high school students). It is indeed important to remark that the gamification principle used in the rogaining teamwork model provides a perfect context in which to embed a heterogeneous set of activities covering basic topics of the bachelor degree (programming, complexity, mathematics, etc) as well as topics and applications related to advanced courses (artificial intelligence, computer security, IoT, etc). In other words, the rogaine competition replaces more traditional orientation activities with a learning by doing experience with a direct contact with other students (i.e., early assistantships) and instructors. The only similar format that can be found in the literature is the Run and Solve competition focused on mathematics as a part of Caucasus Mathematical Olympiad [39].

5.3 Topic Coverage

The checkpoints included in our rogaining events are based on very different technologies and applications ranging from coding and computational thinking to artificial intelligence and virtual reality. For what concerns recruitment activities based on computational thinking and coding, in the literature we can find several important initiatives based on visual languages and human interaction tools [46, 44, 43, 50]. Since our activities assume no prior programming background, some checkpoints propose coding activities relying on visual languages, one of the most difficult points is the selection of an adequate set of building blocks that each player can use in order to specify interactions between sprites in the virtual world or between the human player and the virtual sprites. Scratch [46, 34] and Pocketcode [37] are perfect tools for stimulating creativity while learning coding by examples and experiments. Indeed, Scratch was created to foster computational creativity. Scratch also provides some mechanisms for the multiplayer mode. However, these features are not easy to control, present several limitations (e.g., cloud variables can be used only by expert users) and very few examples are available. Scratch 3.0 has introduced the motion capture blocks that revealed to be a perfect means for collaborative activities for group of students and that we used in one checkpoint to create competitions between student teams standing in front of a large screen showing a Scratch 3.0 app in which participants interact with sprites. In addition to all the features provided by Scratch, Pocketcode provides blocks for controlling games using sensor data providing additional tangible experience to lab participants. We exploit this feature in the Pocketcode app to be realized in a checkpoint. Concerning the coding language (shapes) adopted in the checkpoint on tangible coding, we took inspiration by approaches on teaching principles of coding for very young children proposed in [18, 24, 48]. In the Virtual Reality checkpoint, the activity has been designed in a simplified virtual world (a single room) in order to avoid problems such as simulation sickness and loss of immersiveness [29] (see also [56, 51, 53, 52]) that however are challenging problems for more complex VR applications.

Although each checkpoint activity has innovative aspects with respect to coding laboratories that we are aware of, we remark that the novelty of our proposal is the entire model in which rogaining is used as a perfect means to integrate very different computer science areas and applications that students will encounter in their future career. Furthermore, the model and the supporting tools such as the SR-App can be easily adapted to other disciplines by modifying the activities in the checkpoints.

5.4 Soft Skills

The importance of teamwork for computer science orientation had been recognized very soon [12], and, in recent years activities based on the development of teamwork and social interaction skills are gaining more and more attention [32].

The proposed format, differently from scavenger hunt [54], requires some form of planning and goal prioritization, since teams are forced to select a strict subset of checkpoints to visit during the competition, and of dynamic re-planning (reaction to unexpected events) in case the planned checkpoint activity is occupied by another team. Thus, our format allows participants to train planning, goal prioritization, time management, and negotiation skills. The use of gamification for soft skills such as planning and goals prioritization has been recently proposed for to-do list management in [11].

6 CONCLUSIONS AND FUTURE DIRECTIONS

In this paper we presented an innovative teamwork format for short activities (4/5 hours) offered during internships for high school students at our University. The format has been applied in many different occasions and with different configurations, starting from 2019. The format has been designed considering important challenges in computer science orientation and education such as coverage of the wide range of topics of our discipline (using the different activities proposwed in the checkpoints), the need of acquiring both technical and soft skill (e.g. strategy and team work during the rogaine), and the need of reducing gender gap and improving inclusion (combining soft skills and activities related to several different application domains such as data analysis, IoT, healthcare, etc).

Our format can be easily customized by adding other type of activities to improve coverage. Indeed, checkpoints may range from more theoretical to more practical ones. Furthermore, the small number of members in teams allows us to optimize the use of specific hardware resources such as AR/VR visors, tablets, and devices. Some of our students were involved in the design of the software and hardware used in the activities. This provides an additional clear example of technical skills that could be acquired by studying computer science. Our students were also involved in the design of checkpoints, achieving a further meta-goal besides the orientation one: showing students how to convey computer science concepts and methods in a limited amount of time and in an engaging way.

In principle our format and web app can be applied to other disciplines by changing the contents (or the goals) of the checkpoints.

The activity turned out to be appreciated by the participants, and, together with the internship program it is part of, to be effective in promoting informed enrollment to our degree, as shown by the analysis reported in Section 4.

After the COVID-19 pandemic, the challenge we are facing is the preparation of activities in blended learning with both students in presence and in remote. We have recently organized a first experiment in this direction dividing 120 high school interns in two equally distributed groups. Both groups attended the same presentations and the same laboratories proposed in the two versions. We have collected data for both groups and we plan to carry out a detailed evaluation the experience in order to compare the performance of the different activities using the same model adopted for the rogaining events.

CONFLICT OF INTEREST STATEMENT

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

AUTHOR CONTRIBUTIONS

G.D. G.G., L.G., and V.M. are the main designers of the rogaine format. V.M. worked on the format workflow specification. F.V. supported the team formation process. A.F. is the main author of the SR-App and related back-end software. M.C., G.D., G.G., F.O., N.N have designed the challenges used in the 2019 and 2020 rogaine editions. All the authors contributed to writing and editing.

FUNDING

G. Delzanno,G. Guerrini, and V. Mascardi were supported by the Boosting Computational Thinking with Pervasive and Collaborative Technologies 2017-2019 Grant funded by the University of Genoa.

ACKNOWLEDGMENTS

The authors would like to thank all students who helped the authors in conducting the different editions of the team-based activity.

SUPPLEMENTAL DATA

Additional data are available on Github at the URL `https://bit.ly/smartrogaining`

INFORMED CONSENT

Written informed consent was obtained from the individual(s) and minor(s)' next of kin for the publication of any potentially identifiable images or data included in this article.

REFERENCES

[1]C. Aritajati, M. B. Rosson, J. Pena, D. Cinque, and A. Segura. A socio-cognitive analysis of summer camp outcomes and experiences. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, pages 581–586, 2015.

[2]R. F. Bales. *Interaction process analysis; a method for the study of small groups.* Addison-Wesley, 1950.

[3]G. Ballestin, C. Bassano, F. Solari, and M. Chessa. A virtual reality game design for collaborative team-building: A proof of concept. In Tsvi Kuflik, Ilaria Torre, Robin Burke, and Cristina Gena, editors, *Adjunct Publication of the 28th ACM Conference on User Modeling, Adaptation and Personalization, UMAP 2020, Genoa, Italy, July 12-18, 2020*, pages 159–162, New York , NY, USA, 2020. ACM.

[4]A. K. Barianos, A. Papadakis, and N. Vidakis. Content manager for serious games: Theoretical framework and digital platform. *Advances in Mobile Learning Educational Research*, 2(1):251–262, 2022.

[5]I. Blecic, A. Cecchini, and V. A. Colorni. University explorer: a web-based role-play game for university orientation. In *Learning with Games International Workshop*, pages 337–342, 2007.

[6]M. E. Caspersen, J. Gal-Ezer, A. D. McGettrick, and E. Nardelli. Informatics for all: The strategy, acm europe and informatics europe, 2018.

[7]M. E. Caspersen, J. Gal-Ezer, A. D. McGettrick, and E. Nardelli. Informatics as a fundamental discipline for the 21st century. *Commun. ACM*, 62(4):58, 2019.

[8]Catrobat pocketcode, 2021. `https://share.catrob.at/app/`. Accessed on 2021.

[9]S. Cheryan, S. A. Ziegler, A. K. Montoya, and L. Jiang. Why are some stem fields more gender balanced than others? *Psychological bulletin*, 143(1):1, 2017.

[10]Code.org. Hour of code. `https://code.org`.

[11]S. Consul, J. Stojcheski, V. Felso, F. Lieder, R. He, A. Mehta, Y. R. Jain, A. Kemtur, and M. Tosic. Leveraging ai for effective to-do list gamification. In *5th international workshop Gam-R – Gamification Reloaded*, 2022.

[12]C. R. Cook. Cs0: Computer science orientation course. In *Proceedings of the twenty-eighth SIGCSE technical symposium on Computer science education*, pages 87–91, 1997.
[13]G. Delzanno, G. Guerrini, V. Mascardi, L. Gelati, V. Petito, F. Vitali, A. Ferrando, D. Ancona, M. Chessa, N. Noceti, and F. Odone. Slow rogaining: An innovative teamwork model for computer science education. In T. Kuflik, I. Torre, R. Burke, and C. Gena, editors, *Adjunct Publication of the 28th ACM Conference on User Modeling, Adaptation and Personalization, UMAP 2020, Genoa, Italy, July 12-18, 2020*, pages 119–126, New York , NY, USA, 2020. ACM.
[14]G. Delzanno, G. Guerrini, S. Perotto, F. Traverso, S. Mammoliti, and S. Staglianó. Codinji: Welcome to the coding jungle! In T. Kuflik, I. Torre, R. Burke, and C. Gena, editors, *Adjunct Publication of the 28th ACM Conference on User Modeling, Adaptation and Personalization, UMAP 2020, Genoa, Italy, July 12-18, 2020*, pages 151–154, New York , NY, USA, 2020. ACM.
[15]E.Vegas, M. Hansen, and B. Fowler. Building skills for life: How to expand and improve computer science education around the world, centre fdor universal education at brookings, 2021.
[16]Z. Fitz-Walter, D. Tjondronegoro, and P. Wyeth. A gamified mobile application for engaging new students at university orientation. In *Proceedings of the 24th Australian Computer-Human Interaction Conference*, pages 138–141, 2012.
[17]M. Segers F.J. Dochy and D. Sluijsmans. The use of self-, peer and co-assessment in higher education: A review. *Studies in Higher education*, 24(3):331–350, 1999.
[18]G. Futschek and J. Moschitz. Learning algorithmic thinking with tangible objects eases transition to computer programming. In *ISSEP*, pages 155–164, 2011.
[19]W. Gander, A. Petit, G. Berry, B. Demo, J. Vahrenhold, A. McGettrick, R. Boyle, M. Drechsler, A. Mendelson, C. Stephenson, C. Ghezzi, and B. Meyer. Informatics education: Europe cannot afford to miss the boat, report of the joint informatics europe and acm europe working group on informatics education, 2013.
[20]M. R. R. Gari, G. S. Walia, and A. D. Radermacher. Gamification in computer science education: A systematic literature review. In *2018 ASEE Annual Conference & Exposition*, 2018.
[21]M. N. Giannakos, I. O. Pappas, L. Jaccheri, and D. G. Sampson. Understanding student retention in computer science education: The role of environment, gains, barriers and usefulness. *Education and Information Technologies*, 22(5):2365–2382, 2017.
[22]S. Glencross and S. Elsom. Using an alternate reality game to facilitate student engagement during orientation. *Student Success*, 10(2):13–23, 2019.
[23]L. Happe, B. Buhnova, A. Koziolek, and I. Wagner. Effective measures to foster girls' interest in secondary computer science education. *Educ. Inf. Technol.*, 26(3):2811–2829, 2021.
[24]M. Horn and R. Jacob. Designing tangible programming languages for classroom use. In *1st international conference on tangible and embedded interaction*, pages 159–162, 2007.
[25]A. Hutzler, R. Wagner, J. Pirker, and C. Gütl. Mythhunter: Gamification in an educational location-based scavenger hunt. In D. Beck, C. Allison, L. Morgado, J. Pirker, F. Khosmood, J. Richter, and C. Gütl, editors, *Immersive Learning Research Network - Third International Conference, iLRN 2017, Coimbra, Portugal, June 26-29,*

2017, Proceedings, volume 725 of *Communications in Computer and Information Science*, pages 155–169, Heidelberg, 2017. Springer.

[26]K. Kori, M. Pedaste, and O. Must. The academic, social, and professional integration profiles of information technology students. *ACM Trans. Comput. Educ.*, 18(4):20:1–20:19, 2018.

[27]K. Kori, M. Pedaste, E. Tonisson, T. Palts, H. Altin, R. Rantsus, R. Sell, K. Murtazin, and T. Ruetmann. First-year dropout in ICT studies. In *IEEE Global Engineering Education Conference, EDUCON 2015, Tallinn, Estonia, March 18-20, 2015*, pages 437–445. IEEE, 2015.

[28]C. Lang, A. Craig, and M.A. Egan. The importance of outreach programs to unblock the pipeline and broaden diversity in ict education. *International Journal of Information and Communication Technology Education (IJICTE)*, 12(1):38–49, 2016.

[29]J. LaViola. A discussion of cybersickness in virtual environments. *ACM Sigchi Bulletin*, 32(1):47–56, 2000.

[30]F. Lazarinis, I. Boididis, L. Kozanidis, and D. Kanellopoulos. An adaptable multi-learner serious game for learning cultural heritage. *Advances in Mobile Learning Educational Research*, 2(1):201–215, 2022.

[31]M. Leotta, V. Petito, L. Gelati, G. Delzanno, G. Guerrini, and V. Mascardi. Orchestrated crowdsourced testing of a mobile web application: a case study. In *Conference Companion of the 3rd International Conference on Art, Science, and Engineering of Programming, Genova, Italy, April 1-4, 2019*, pages 17:1–17:6, New York , NY, USA, 2019. ACM.

[32]R. Lingard and S. Barkataki. Teaching teamwork in engineering and computer science. In *2011 Frontiers in Education Conference (FIE)*, pages F1C–1. IEEE, 2011.

[33]J. B. Main and C. Schimpf. The underrepresentation of women in computing fields: A synthesis of literature using a life course perspective. *IEEE Transactions on Education*, 60(4):296–304, 2017.

[34]J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond. The scratch programming language and environment. *ACM Trans. Comput. Educ.*, 10(4):16:1–16:15, 2010.

[35]A. Manzoor, M. Samarin, D. Mason, and M. Ylianttila. Scavenger hunt: Utilization of blockchain and iot for a location-based game. *IEEE Access*, 8:204863–204879, 2020.

[36]M. Marzolla and R. Mirandola. Gender balance in computer science and engineering in italian universities. *CoRR*, abs/1907.07009, 2019.

[37]M. Müller, C. Schindler, and W. Slany. Pocket code: a mobile visual programming framework for app development. In E. Tilevich, editor, *Proceedings of the 6th International Conference on Mobile Software Engineering and Systems, MOBILESoft@ICSE 2019, Montreal, QC, Canada, May 25, 2019*, pages 140–143. IEEE / ACM, 2019.

[38]MUR and CINI. Programma il futuro. `https://programmailfuturo.it/`.

[39]D. V. Musatov, M. I. Kalina, O. N. Malkhozheva, A. V. Yurov, and D. K. Mamiy. Run and solve: Competitions in mathematical rogaining. *Russ. Digit. Libr. J.*, 22(6):672–685, 2019.

[40]N. Noceti, F. Odone, A. Marsella, M. Moro, and E. Nicora. Tangible coding for kids with AI inside. In Tsvi Kuflik, Ilaria Torre, Robin Burke, and Cristina Gena, editors, *Adjunct Publication of the 28th ACM Conference on User Modeling, Adaptation and*

Personalization, UMAP 2020, Genoa, Italy, July 12-18, 2020, pages 163–166, New York , NY, USA, 2020. ACM.

[41]M. Ohland, M. Loughry, D. J. Woehr, L. G. Bullard, R. M. Felder, C. J. Finelli, R. A. Layton, H. R. Pomeranz, and D. G. Schmucker. The comprehensive assessment of team member effectiveness: Development of a behaviorally anchored rating scale for self-and peer evaluation. *Academy of Management Learning & Education*, 11(4):609–630, 2012.

[42]S. Papadakis. Evaluating a game-development approach to teach introductory programming concepts in secondary education. *International Journal of Technology Enhanced Learning*, 12(2):127–145, 2020.

[43]H. Partovi. Transforming US education with computer science. In J. D. Dougherty, K. Nagel, A. Decker, and K. Eiselt, editors, *The 45th ACM Technical Symposium on Computer Science Education, SIGCSE '14, Atlanta, GA, USA - March 05 - 08, 2014*, pages 5–6, New York , NY, USA, 2014. ACM.

[44]H. Partovi and M. Sahami. The hour of code is coming! *SIGCSE Bulletin*, 45(4):5, 2013.

[45]G. N. Phillips and R. Phillips. *Rogaining: cross-country navigation.* Outdoor Recreation in Australia, Melbourne, 2000.

[46]M. Resnick, J. Malone, A. Monroy-Hernandez, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. S. Silver, B. Silverman, and Y. B. Kafai. Scratch: programming for all. *Commun. ACM*, 52(11):60–67, 2009.

[47]M. B. Rosson, J. M. Carroll, and H. Sinha. Orientation of undergraduates toward careers in the computer and information sciences: Gender, self-efficacy and social support. *ACM Transactions on Computing Education (TOCE)*, 11(3):1–23, 2011.

[48]S.Cooper, W. Dann, and R. Pausch. Teaching objects-first in introductory computer science. In *Proceedings of the 34th SIGCSE technical symposium on Computer science education*, pages 191–195, 2003.

[49]Scratch 3.0, 2021. `http://scratch.mit.edu/`. Accessed on 2021.

[50]S. Sentance, J. Waite, S. Hodges, E. MacLeod, and L. Yeomans. Creating cool stuff: Pupils' experience of the BBC micro: bit. In M. E. Caspersen, S. H. Edwards, T. Barnes, and D. D. Garcia, editors, *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education, Seattle, WA, USA, March 8-11, 2017*, pages 531–536, New York, NY, USA, 2017. ACM.

[51]M. Slater, A. Steed, and M. Usoh. The virtual treadmill: A naturalistic metaphor for navigation in immersive virtual environments. In *Virtual Environments 95*, pages 135–148. Springer, Heidelberg, 1995.

[52]F. Steinicke, Y. Visell, J. Campos, and A. Lécuyer. Human walking in virtual environments. *Perception, Technology, and Applications*, 56(7), 2013.

[53]E. Suma, S. Clark, D. Krum, S. Finkelstein, M. Bolas, and Z. Warte. Leveraging change blindness for redirection in virtual environments. In *2011 IEEE Virtual Reality Conference*, pages 159–166. IEEE, 2011.

[54]J. O. Talton, D. L. Peterson, S. Kamin, D. Israel, and J. Al-Muhtadi. Scavenger hunt: computer science retention through orientation. In Doug Baldwin, Paul T. Tymann, Susan M. Haller, and Ingrid Russell, editors, *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education, SIGCSE 2006, Houston, Texas, USA, March 3-5, 2006*, pages 443–447, New York , NY, USA, 2006. ACM.

[55]J. M. Wing. Computational thinking. *Commun. ACM*, 49(3):33–35, 2006.
[56]F. Zhang, S. Chu, R. Pan, N. Ji, and L. Xi. Double hand-gesture interaction for walk-through in vr environment. In *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, pages 539–544. IEEE, 2017.

1 WORKFLOW SPECIFICATION OF THE ROGAINING ACTIVITY

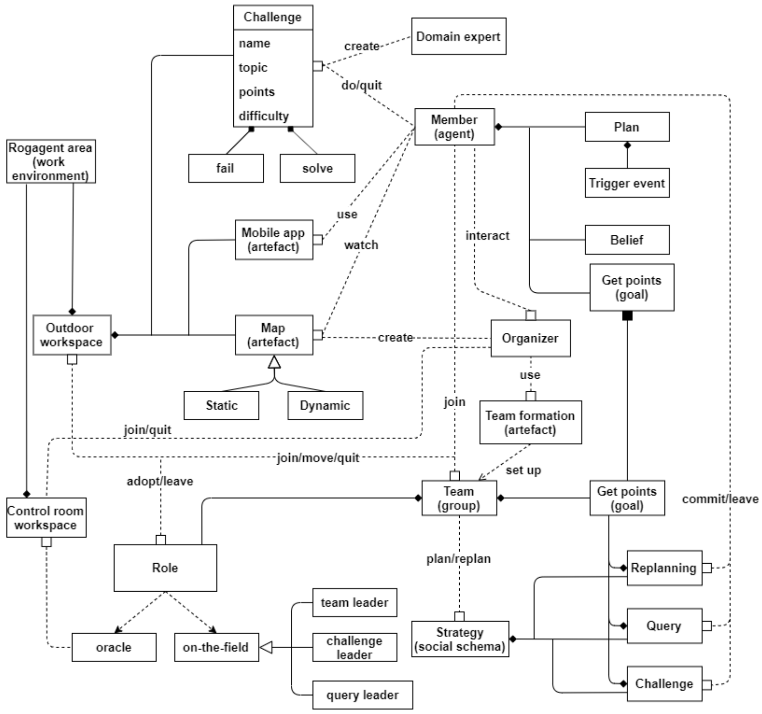

Figure 6. Slow Rogaining architecture.

Fig. 6 shows an UML-like diagram of the workflow associated to the team-based rogaining activity described in the paper.

In the UML scheme the rogaining scenario is divided in two workspaces: the *outdoor workspace* where the on-the-field team members move through checkpoints and where domain experts organize their challenges, and the *control room* for organizers and team coordinators, the latter ready to advise and interact with their teammates.

The team formation is a fundamental stage and is supposed to take place in a preliminary preparation phase. Each team is equipped with a map of the area. The map can be either *static* (can be read by anyone and shows where challenges are located and their peculiar information) or *dynamic*, providing dynamic and updated information on where the teams

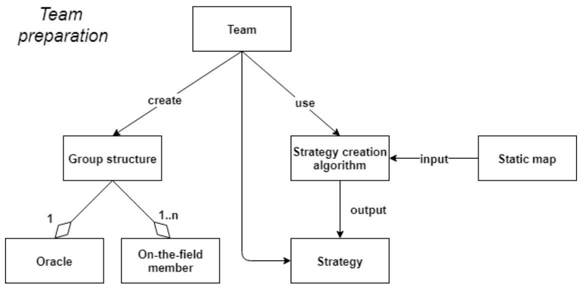

Figure 7. Slow Rogaining team preparation.

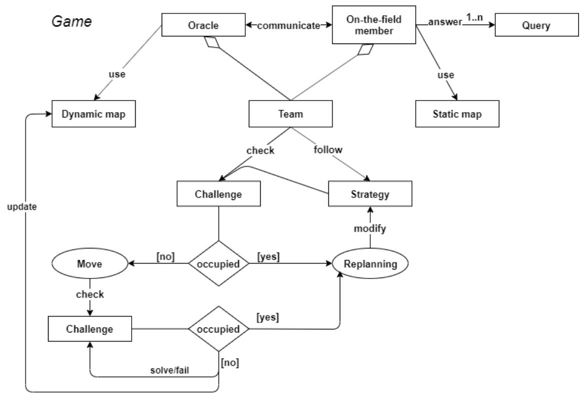

Figure 8. Slow Rogaining game.

are located and the status of each challenge (free or occupied). The dynamic map is an extension of the static one, but it can be accessed only by people in the control room workspace. *Challenges* are created by domain experts who define their features such as the topic of the challenge, the points that can be earned by solving it and the difficulty. Each challenge takes place in a different location of the event area - inside the outdoor workspace. Team members can face the challenge only when they physically reach its location. The general scheme also considers the possibility of using the SR-App to submit quizzes to the different teams during the rogaine, e.g., to provide additional information on the proposed laboratories or to simply to increase the difficulty level of the challenge.

The domain expert is responsible for creating the various challenges according to the theme of the event. Each team is uniquely identified via the team credentials used to access SR-App. The idea behind splitting inside a team is not a physical, but only a logical, role-based one, for better shaping ‘who does what’ based on the participants attitudes. Sub-roles of ‘on-the-field’ members can be for example the *team leader*, the *challenge leader*, and the *query leader*. The team leader emerges during the game progress according to how the members behave and interact with others. The other two roles are also assigned at runtime and can be adopted, possibly by different members, each time a team reaches a challenge. The *on_the_filed* team members other than the current challenge/query leaders will support either the resolution of the challenge or query answering. The last role is the team coordinatore. She/he has a complete view of the situation on the field and can suggest the next move when replanning is needed. The main stages of the process, which are depicted in Figures 7 and 8. We do not include briefing and the debriefing stages. The organisers set up the rogaining event by exploiting some team formation algorithm and by communicating to each participant the team they belong to. Once everything is set, each team has some time to consult the map, discuss and exploit some strategy creation algorithm in order to choose jointly the best strategy to follow. At that time, in each team an initial internal structure, with roles associated with participants, is arranged. Each team waits for the three-two-one-go command of the organizer to then move to the first challenge chosen according to its strategy. In the event that such challenge is occupied the team moves to one of those not faced yet. As shown in Figure 8, when a team has reached a challenge there are two possibilities: to solve or to fail.