Challenges Implementing the SimProgramming Approach in Online Software Engineering Education for Promoting Self and Co-regulation of Learning

Daniela Pedrosa
Universidade de Aveiro & CIDTFF, *Aveiro & Universidade de Trás-os-Montes e Alto Douro,*
Vila Real, Portugal
dpedrosa@ua.pt
0000-0001-9536-4234

Leonel Morgado
Universidade Aberta & INESC TEC, Coimbra, Portugal
leonel.morgado@uab.pt
0000-0001-5517-644X

José Cravino
Universidade de Trás-os-Montes e Alto Douro, Vila Real & CIDTFF,
Aveiro, Portugal
0000-0002-5376-6128

Mario Madureira Fontes
Department of Computer, Pontifícia Universidade Católica de São Paulo,
São Paulo, Brazil
0000-0002-5618-0616

Maria Castelhano
Department of Education and Psychology,
Universidade de *Aveiro,*
Portugal.
mfmcastelhano@live.ua.pt

Claudia Machado
Universidade de Trás-os-Montes e Alto Douro, Vila Real & CIDTFF – Research Centre on Didactics and Technology in the Education of Trainers,
Aveiro, Portugal
0000-0001-9132-3240

Eliana Curado
Department of Education and Psychology,
Universidade de Aveiro,
Portugal
eliana13@ua.pt

***Abstract*—High academic failure rates in computer programming are significant transitioning from initial to advanced stages. In online higher education, challenges are greater since students' autonomy requires greater skills for self-regulation and co-regulation of learning. The SimProgramming approach develops these skills and is being adapted to e-learning for this transitioning phase. In this paper, we describe the dynamics and outcomes of student participation and task development in a first iteration of the adapted e-SimProgramming approach, which took place during a 2nd year-2nd semester course for the Informatics Engineering program at Universidade Aberta in the 2018/2019 academic year. We identified pedagogical and technical challenges, requiring changes for subsequent attempts of adopting SimProgramming for online education contexts: target audience and teaching context aspects; self and co-regulation of learning dimensions of e-learning courses; pedagogical design recommendations; and requirements for software tools for learning management.**

***Index Terms*—Computer Programming, e-SimProgramming approach, e-learning, Self and Co - Regulated Learning,**

I. INTRODUCTION

There is handcrafting, and there is engineering. Medieval builders built huge cathedrals by painfully, gradually, adding small improvements to existing approaches - often failing catastrophically. Modern civil engineers plan entirely new approaches, in novel conditions, while attesting confidence in their subsequent structural success. Similarly, software engineering students learning computer programming start by learning it as a craft. But the goal of software engineering is not to educate master craftspeople: it is to educate engineers. Software engineers should master their programming craft but also acquire engineering skills and competences to produce works with qualities such as resilience, maintainability, and adaptability - even when tackling novel and distinct problems.

This combination of programming crafts with the engineering of programming strikes students in higher education when transitioning from novice to advanced programming, a particularly difficult moment, with high rates of academic failure [1]. This transition is the focus of the SimProgramming approach [2], supporting the development of students' skills for self and co-regulation of learning (SCRL) as a way to overcome this challenging transition, advocating problem-based learning activities, as teams, over four phases with specific tasks of varying durations [3].

In distance learning (DL), SCRL plays a core role. However, that SimProgramming approach was developed within the context of face-to-face higher education [2]. Its application to DL is not straightforward. The main challenges are: a) DL is mainly asynchronous, providing working students with flexibility to manage their study time and emphasizing autonomy - hence activities must be possible entirely without synchronous contact with lecturers or colleagues; b) few opportunities for casual, serendipitous contacts for feedback and tracking, hence these must be planned for as part of the course; c) the student population has a higher average age, typically employed and with extended family responsibilities, which renders teamwork approaches challenging, by complicating coordination and availability; d) feedback loopback time is longer, because study time takes place outside typical working hours or working days, when lecturers are not available for direct inquiries (and often institutional DL models involve some gap time between feedback requests and its

978-1-7348995-0-4/20/$31.00 ©2020 Immersive Learning Research Network

provision). These contextual distinctions require more discipline and autonomy from students (SCRL skills). Since these are the focus of the SimProgramming approach [2], employing it in DL warrants exploration. Thus, we are reformulating SimProgramming for DL: we are creating the e-SimProgramming approach.

To convert SimProgramming into e-SimProgramming, the areas of focus for a first research iteration (see section IV), were: the new context dynamics (asynchronous); the supporting technologies (learning management system); target audience (30-50 year-olds); pedagogical preferences of the teacher for field trials (project-based learning instead of problem-based learning); time constraints for students and lecturer; & administrative & regulatory differences (requirement of choice between assessment paths, mandatory restriction of 40% maximum impact of project-based activities in the final grade). In this paper, we describe the identified pedagogical and technical challenges: low and late student participation, students' sense of isolation while interacting in forum tasks and low perception of class dynamics; the need for distinct feedback mechanisms (automated and lecturer); misunderstandings of text-based task descriptions. From these outcomes, we present recommendations for the adaptation of SimProgramming to DL in subsequent iterations.

II. BACKGROUND

Transitioning from entry-level programming to advanced programming is laden high rates of academic failure [3]. Reasons include inadequate teaching approaches, lack of student involvement, inadequate student learning strategies, lack of teamwork and cooperation skills, difficulties with abstract approaches to data organization and program control, such as architectural styles (such as Model–View–Controller, or MVC), and other software engineering concepts.

Computer programming DL itself brings specific challenges, identified in a systematic survey [5]: a) different requirements upon individual characteristics, such as SCRL technical skills; b) older students, with more professional and family responsibilities, requiring better time management, and increasing chances of work overload; c) e-pedagogy practice: low interaction with tutors and among students, feeling of isolation, lack of appropriate technical requirements, inadequate learning resources, and demanding syllabus [4]. These challenges can be overcome through pedagogical strategies, such as providing guidance/scripts and continuous feedback that facilitate the process of planning and managing learning, and problem-solving skills. The creation of flexible learning environments, development of solutions in the face of a certain context, and adoption of appropriate assessment tools contribute to better optimization of adult students' learning and to avoid students dropping out of the course [4].

The literature provides recommendations: involve students by raising questions and promoting debate; provide timely, constructive feedback; reflect upon and adjust their pedagogical practices towards enriching students' learning; upon course start, provide rules and policies to frame students' expectations; perform formative assessment as an engagement strategy [5]. Other relevant aspects [6] are the content delivery form, media type, student skillset, and syllabus content. Also, monitoring students' progress helps minimize the likelihood of dropout and increases participation [7]. Lectures should be interactive and attractive to students, and learning progress monitoring can be done, for instance, with short tests with immediate automated feedback [9], beneficial to students reckoning of their level of understanding and study progress. Role-playing business environments helps students to immerse themselves in situations similar to the real world: develop projects and respond to problems through active learning [11]. They experience narrative immersion and challenge-based immersion [8].

Most computer programmers do not realize the importance of SRL skills for their learning process and achieving success. CRL skills help students improve their programming skills, providing resources and skills to work with others [9]. In DL courses, challenges are greater and dropout rate is higher [10], with students exhibiting difficulties planning, developing, and using SCRL skills adequately [11]–[13]. Thus, there is potential lack of immediate support and risk of social isolation [14], being necessary to explore SCRL-promoting pedagogical strategies, improve their personal learning environments, and clarify what support is most appropriate to each student [15]. A framework for promoting SRL skills is proposed in the literature [16] with eight features: learning plan, records/e-portfolio and sharing, evaluation, the type of feedback, scaffolding, agents, and visualization of goals/procedures/concepts. Promoting CRL encourages SRL in adult students, helping them monitor the learning process to develop metacognitive skills and achieve learning goals, facilitated by discussion and reflection spaces [17]

III. TEACHING CONTEXT AND LEARNING ASSIGNMENT

The first field attempt to implement the SimProgramming approach in DL occurred in the 2018/2019 academic year, in the course "Software Development Laboratory" (LDS, Portuguese-language acronym). The format was online & asynchronous, in the Moodle platform of Universidade Aberta (UAb), Portugal, during the 2nd semester of the 2nd year of the Informatics Engineering baccalaureate programme, over 12 academic weeks. The course goal is to scaffold undergraduates transitioning from novice programmers into proficient programmers, over a six-topic syllabus.

The student cohort is heterogeneous in age (24 to 55 years old), gender, location (different regions of the country and abroad), and educational level (a mix of secondary education, some college attendance, and undergraduate degrees in fields other than computing), but already part of the workforce.

Students can choose from one of two available assessment paths: continuous assessment (assignments, called e-folios, plus p-folio written test) or final written exam. In continuous assessment, students pursue project-based learning activities either as part of a team or individually. In the 2018/2019 academic year, of the 58 enrolled students, 8 chose the final exam and 50 the continuous assessment path. Since the course takes place asynchronously, no specific schedule hours exist for the activities. At their own pace, students complete them within their schedule, as long as deadlines are kept. In each topic, activities include forum discussions with the teaching staff and

with colleagues, also asynchronously. The last topic asks students to provide a final project report.

Temporal flexibility is mandated by UAb's pedagogic model [18]. Students' age span and occupations imply scheduling constraints of active professional careers and the need to care for children or older relatives. Being disseminated across the globe, students deal with varying time zones: small teams can arrange to meet, but there is no live lecturing schedule that would fit all. During the two-week span of each topic, teaching must track and encourage progress and interaction.

IV. Differences Between the SimProgramming Approach and the First Iteration of e-SimProgramming

SimProgramming [2] was created in a face-to-face context, from 2011 to 2014, based on four conceptual foundations: (1) business-like learning environment, (2) SRL; (3) CRL; and (4) formative assessment. Teaching strategies stimulate use of SCRL strategies by students, during a course-long assignment [19], with teams and problem-based learning activities, over four stages [3], each with tasks of varying duration where participants role-play roles, immersed in the assignment narrative and challenges.

The first attempt to employ the SimProgramming approach in an DL context took place in the 2015-2016 academic year, as a master's dissertation. It provided insights on the challenges and potential of adapting it to DL: as a motivation and effort regulation instrument; team communication; feedback provision and time; assessment [20]. This inspired the current effort to create its DL version: e-SimProgramming. We focus on these contextual differences: synchronous vs. asynchronous class; Content Management System vs. Learning Management System; young adults vs. older adults; different teaching methods (problem-based vs. project-based learning).

The change in teaching methods was due to lecturer's preference between these inquiry-based approaches. They are similar to the point of being jointly referred to as PBL [21]. The adoption of project-based learning in e-SimProgramming means procedural emphasis. Different problems will occur and teachers typically instruct or coach, whereas in problem-based learning their role is more one of tutoring [22].

Adjusted variables were deadline for task completion, choice of assessment pathway, and the weight of project activities in the final evaluation (40%). This was required to comply with the UAb pedagogical model [23]. Other changes were due to the DL context: elimination of presentations of work by students and status reports. Biweekly reflections [19, 24] were adjusted, now an online form, ceasing to be mandatory.

V. Methodology and Data Collection

An exploratory qualitative methodology was adopted through descriptive analysis [28] of the social interaction phenomena occurring in the Moodle platform: in forums, task submissions, and other items, to illuminate the relationship between the type of learning path chosen by the students and the application of SCRL strategies. The research goals were: 1) understand what pedagogical and technical phenomena occurred during this first iteration; 2) describe the phenomena regarding student participation and use of SCRL strategies; and 3) identify which pedagogical and technical aspects to improve in subsequent iterations of e-SimProgramming.

Descriptive analysis quantified number of forum posts, content addressed (with categories: participants per forum; interventions per participant; interaction type; posts within deadlines), student communication about dropping out and its reasons, and technical/pedagogical problems. Graphs tracked student participation throughout the course, to reveal patterns related to the learning path chosen by students. Interactions were mostly teacher-student or student-student. Collection, treatment and interpretation of data were carried out by two researchers, with validation and reliability via review with the support of two other researchers.

VI. Results and Discussion

A. Analysis of Interactions in the Discussion Forums

Of 50 students choosing continuous assessment in the 2018/2019 academic year, 33 accepted to participate in the research. Of these, 25 chose individual activities; 8 chose team-based activities, forming two teams: Team A, with 5 students, and Team B, with 3. Team A achieved the learning goals. Team B split up at the beginning of the third topic: 1 student quit and the remaining changed to individual activities.

In the first topics (1 & 2) most students completed tasks within the deadline (Fig. 1). For topics 3-5 the opposite occurred: a sharp drop task completion, the majority completing tasks past deadlines. In topic 6, task completion increased, but only 23 students delivered the final report.

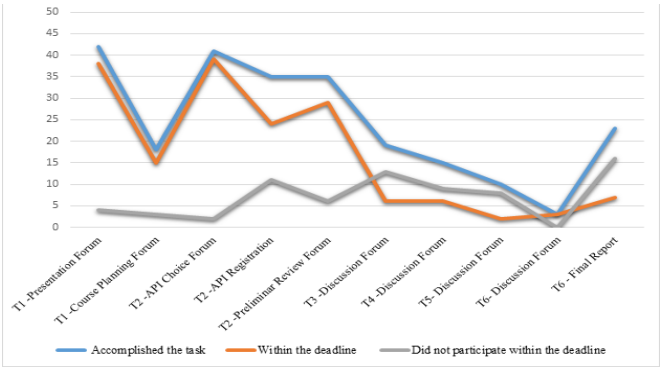

Fig. 1. Analysis of task completion and deadline fulfilment

Course Planning Forum: 18 threads created; 9 with some interaction: 8 with Student-Teacher interaction (STI); just 1 with student-student interaction (SSI).

Topic 2: 30 threads created in the preliminary discussion forum, STI occurred in a single thread, no STI or SSI in the other 29 threads.

Topic 3: 16 threads created: 4 with considerable dynamics, 3 with discussion almost entirely STI. One thread was an internal team discussion. The remaining 12 threads had no discussion dynamics, just plain Post-Response interactions, mostly STI.

Topic 4: 13 threads created. 3 with considerable discussion dynamics but including one where dynamics occurred among

Team A students and between them and the teacher. The other 2 high-dynamics threads were created by 2 students on the individual activities path. The remaining 10 threads exhibited mostly STI.
Topic 5: 9 threads created, only 3 witnessed considerable discussion dynamics, including one with team A participation, and the 2 students in the individual activities path. The other 7 threads witnessed mostly STI, with low participation.
Topic 6: 3 posts created. All exhibited only STI.

In topics 1 and 2 the students participated regularly to accomplish tasks. In topics 3-6 this was no longer the case. There was a decrease in student participation and a gradual tendency to submit assignments after deadlines, throughout the topics, with dropout increasing. The final topic (#6) saw a slight increase due to participation of non-regular students.

There are few occurrences of SSI. Only among groups (teams A & B) did we find some level of interactivity and discussion dynamics. Even though Team B dropped out at an early stage, they had interactions between them until that point. Students on the individual path had mostly STI, no student-student interactivity. In interactions with the teacher, there were few debates. Most interactions were mere question and response.

B. Analysis of interactions in doubt-clearing forums

Over topics the number of threads trended downward. Key inflection moments are from Topic 2 to Topic 3 (matching a transition in assessment - e-folio 1 to e-folio 2) and from Topic 4 to Topic 5 (another assessment transition phase - e-folio 2 to e-folio 3). The topics with most threads were the early ones: topic 1 and topic 2 (5 threads in each).

Overall, the total threads created in the doubt-clearing forums during the course was small (21 for a course of 50 students), with 15 of these being the general doubt clearing forum, which means that the majority of students either had no doubts in the specific topics or did not present them. Of 50 students enrolled in the course, only 13 participated in the doubts forums (either by creating threads or posting in existing ones). Among those participating, the most active were S31 (initially a member of Team B who switched to the individual path), S26 and S28 (members of Team A), and S5 (individual path). All these students had given their consent to participate in this research.

Generic Doubt-Clearing Forum: 18 threads created by 15 of the 50 students. Throughout the semester these covered aspects such as: assessment (e-folio/exam); participation in the research study; reporting a broken link in reading materials. Interactions were mostly STI, but a few SSI occurred.
Topic 1: 3 students created 5 threads with doubts on the C# programming language and on establishment of team B. Interaction took place within the deadline, except in the final thread, with post-deadline interactions. Participation level was low, with this pattern: 1) student created thread; 2) teacher responded; 3) original student replied. Only in a single post did interactivity occur among students (S31, S32 and S33 - students of team B). In all situations, teacher feedback occurred within 24 hours.
Topic 2: 4 students created 5 threads with doubts on: where to provide team meeting logs; resources about APIs; use of APIs in an MVC architectural style; registering APIs used in the course. The first three threads were placed before the stipulated time for activity start. In all threads, interaction was low. Most teacher feedback occurred within 24 hours.
Topic 3: 2 students created 2 threads, within the deadline, asking clarification on the MVC architectural style and code implementation tactics. Interaction in the second of these was richer in duration and number (>5 posts), exceeding the deadline by 9 days, with interaction among team A students and the teacher. In the other, interaction followed the pattern of previous topics.
Topic 4: 4 students created 4 threads (post deadline). Doubts centered on the test implementation phase. In addition to the previous pattern of STI, in thread 1 a student promoted the discussion to the whole class. In this topic, unlike previous ones, teacher feedback occurred within a 48 to 72 hours span.
Topic 5: The teacher created 1 thread encouraging student participation. Students responded asking for clarification about submitting e-folio 3. The teacher answered within 24 hours. No subject matter doubts emerged.
Topic 6: 4 students created 6 threads on the final report and use of the computational concept of C# interface within the MVC architectural style. Interaction followed the customary pattern, with no interaction among students. The teacher responded to students within 24 hours in most instances, often within 48 hours, and rarely exceeding 72 hours. In a few cases the teacher response occurred much later, with acknowledgment that it was due to lack of awareness of its lacking. Periods of breaks in participation tend to match assessment moments: transitions from: e-folio 1 to e-folio 2 and from e-folio 2 to e-folio 3.

C. Overall Analysis

Dropping Out: 3 students on their initiative reported motives for abandoning: *Case 1* (May 2nd - during topic 5): ineffective time management (student enrolled in more courses that she could manage); *Case 2* (May 31st - during topic 6): External locus of control, student informed the teacher that due to pressing job requirements, he hadn't performed tasks for e-folios B/C; *Case 3*, (June 6th - after course finish): External locus of control. Another student informed the teacher that he dropped due to job requirements.

Lack of Students' Perception of Forum Participation Expectations: *Case 4:* After a student comment, the teacher reiterated that forums are "*(...) a space for debate, not dialogue for two. IT IS HIGHLY DESIRABLE for all students here to participate and intervene and exchange ideas (...).*" (Teacher, May 20, 2019, discussion forum of topic 4); *Case 5:* The teacher moved a student's post to the correct forum matching its theme: "(...*) I moved this post to the preliminary analysis forum of the proposed solution, which seems to match it better*." (Teacher, May 7, 2019); and *Case 6:* a student had difficulty finding the forum instructions, so the teacher explained: "*(...) The statement is right at the beginning when you open the page of this forum, so as not to overload the contents of the main course page.* (...)" (Teacher, June 1, 2019).

Lack of Alert Mechanisms for Students and Teacher: The teacher lacked a system warning about not having provided feedback to a particular student, enabling him to monitor the class and individual progress, and check critical points where it is necessary to intervene to lessen drop out risks. E.g., on June 12th the teacher should have given feedback (student post of April 29th with June 1st follow-up - topic 4), "*I'm at fault with you* (...) *I did not give you encouragement feedback* [in that message] (...), *by mere lapse - whenever I do not react within 48 working hours, you should alert me, because something is amiss.* (...)" (Teacher, June 12, 2019). Also, students often missed deadlines, an alert mechanism could help them monitor their deadlines and missing tasks.

Alternate Communication Event (Email): One student presented doubts by email on June 10th (where to submit the final report). We interpret this as a sign of more confidence in the expediency of e-mail over forum posting, for this student.

Delays in Team Creation: In topic 1, six days before the deadline a student questioned the teacher about the creation of teams (a choice of students following the team path) which was going slowly. Few students chose the team path: only two teams were created, one of which quit after some time.

Technical Complaints: In topic 2, on March 23rd, a student reported that an API registration was missing from the repository where he had posted it. This can be an actual technical glitch or a misinterpretation of repository operation, but the exact cause was not ascertained.

VII. LIMITATIONS

The research focused on descriptive analysis and data interpretation, which although reliability has been ensured through peer review of project researchers, it is necessary in future work to triangulate data and perform statistical analysis.

VIII. CONCLUSIONS AND FINAL THOUGHTS

The identified decrease in participation and a gradual failure to keep deadlines, as well as the dropout and social isolation of students, corroborate reported challenges in the literature [4]. Students submitting only in the final topic shows that they have difficulties adopting adequate SRL and CRL skills, as expected from the literature [14], thus for this course and contents pedagogic strategies are necessary to develop such skills. The limited or null student-student interactions in the forums is not an uncommon challenge in the literature [4], [6], [30]. e-SimProgramming improvement requires adequate pedagogical strategies to stimulate this interaction, in particular to encourage working in teams.

The infrequent doubt posts by students reaffirms as necessary to establish appropriate strategies that help students become aware of their difficulties [14]. Against our expectations, of increasing doubts near assessment periods, we saw breaks in participation. It is required to establish the reasons for low level of course/content participation. Possible factors include: difficulty exposing doubts due to "shame" (fear of exposure of lack of understanding to the class and the teacher); difficulty expressing the doubts objectively; lack of interest in contents; preference for other communication channels (e-mail to the teacher, class chat); preference for community clarification (online communities, colleagues, etc.); SRL issues of study time and study organization (no doubts due to no studying).

The teacher overall fast turnaround time, but occasional long delays due to lack of awareness shows that it is difficult for the teacher to perceive class dynamics. This may lead to students feeling isolated due to the lack of interaction [4]. The cases of lack of students' perception of forum participation expectations point towards misunderstandings interpreting the requested tasks. It is necessary to adopt other types of strategies that clarify task instructions to the students.

e-SimProgramming should consider support tools to provide adequate teacher awareness, e.g.: 1) provide visualization of course dynamics to identify students in need of specific support, critical dropout risk periods, breaks in task development or submission. 2) Provide action reminders linked to the monitoring process for teacher and students, providing alerts on due delivery dates, ongoing and missing tasks, and overall monitoring of their learning progress. We are approaching this tool design with a formal analysis of interactions, using BPMN modelling [32].

Also, immersion should be improved alongside its various dimensions: digital narratives, construction of 3D virtual environments, social active learning. Pedagogical strategies should address SCRL development, so students are better able to effectively plan, organize and manage their learning process and create optional moments of socialization.

In future work, analysis of student posts could identify are shortcomings in actual provision of course materials. Learning and reading comprehension difficulties could lead to improvement of materials, to verify which SCRL aspects can be prioritized. We intend to analyze the contribution of these processes and tools to the teaching-learning of programming and the potential for other fields in e-learning.

ACKNOWLEDGMENT

Daniela Pedrosa wishes to thank Fundação para a Ciência e Tecnologia (FCT) and CIDTFF (UID/CED/00194/2019) - Universidade de Aveiro, Portugal, for Stimulus of Scientific Employment – CEECIND/00986/2017 Individual Support 2017. This work is financially supported by National Funds through FCT – Fundação para a Ciência e a Tecnologia, I.P., under project PTDC/CED-EDG/30040/2017.

We would like to thank all the students and teachers who collaborated on this research.

REFERENCES

[1] L. Morgado, et al.: Social networks, microblogging, virtual worlds, and Web 2.0 in the teaching of programming techniques for software engineering: A trial combining collaboration and social interaction beyond college. In: IEEE Global Engineering Education Conference, EDUCON (2012)

[2] D. Pedrosa, et al.: SimProgramming : the development of an integrated teaching approach for computer programming in higher education. In: INTED2016 Proceedings - 10th International Technology, Education and Development Conference March 7th-9th, 2016 — Valencia, Spain. pp. 7162–7172. IATED Academy, Valencia, Spain (2016)

[3] D. Pedrosa, et al.: Self-regulated learning in higher education: strategies adopted by computer programming students when supported by the

SimProgramming approach. Production. 27, (2017). https://doi.org/10.1590/0103-6513.225516
[4] M. Kara, et al.: Challenges Faced by Adult Learners in Online Distance Education: A Literature Review. Open Prax. 11, 5–22 (2019). https://doi.org/10.5944/openpraxis.11.1.929
[5] M. Kebritchi, et al.: Issues and Challenges for Teaching Successful Online Courses in Higher Education: A Literature Review. J. Educ. Technol. Syst. 46, 4–29 (2017). https://doi.org/10.1177/0047239516661713
[6] W.W. Fish, L.E Wickersham, Best practices for online instructors: Reminders. Q. Rev. Distance Educ. 10, 279 (2009)
[7] C. Roddy, et al.: Applying Best Practice Online Learning, Teaching, and Support to Intensive Online Environments: An Integrative Review. Front. Educ. 2, 59 (2017). https://doi.org/10.3389/feduc.2017.00059
[8] S, Agrawal, et al.: Defining Immersion: Literature Review and Implications for Research on Immersive Audiovisual Experiences. In: 147th AES Pro Audio International Convention. p. 14. Audio Engineering Society, New York (2019)
[9] C.W. Tsai, Exploring the effects of online team-based learning and co-regulated learning on students' development of computing skills. Interact. Learn. Environ. 24, 665–680 (2016). https://doi.org/10.1080/10494820.2014.917106
[10] J. Bowers, P. Kumar, Students' Perceptions of Teaching and Social Presence: A Comparative Analysis of Face-to-Face and Online Learning Environments. Int. J. Web-Based Learn. Teach. Technol. 10, 27–44 (2015). https://doi.org/10.4018/ijwltt.2015010103
[11] M.H. Cho, D. Shen, Self-regulation in online learning. Distance Educ. 34, 290–301 (2013). https://doi.org/10.1080/01587919.2013.835770
[12] J.C.Y. Sun, R. Rueda, Situational interest, computer self-efficacy and self-regulation: Their impact on student engagement in distance education: Student engagement in distance education. Br. J. Educ. Technol. 43, 191–204 (2012). https://doi.org/10.1111/j.1467-8535.2010.01157.x
[13] S. Järvelä,, et al.: Enhancing socially shared regulation in collaborative learning groups: designing for CSCL regulation tools. Educ. Technol. Res. Dev. 63, 125–142 (2015). https://doi.org/10.1007/s11423-014-9358-1
[14] N. Dabbagh, A. Kitsantas, Personal Learning Environments, social media, and self-regulated learning: A natural formula for connecting formal and informal learning. Internet High. Educ. 15, 3–8 (2012). https://doi.org/10.1016/j.iheduc.2011.06.002
[15] J. Broadbent, Comparing online and blended learner's self-regulated learning strategies and academic performance. Internet High. Educ. 33, 24–32 (2017). https://doi.org/10.1016/j.iheduc.2017.01.004
[16] M.H. Yen, et al.: A framework for self-regulated digital learning (SRDL). J. Comput. Assist. Learn. 34, 580–589 (2018). https://doi.org/10.1111/jcal.12264
[17] J. Kaplan, Les stratégies d'autorégulation collective des apprenants adultes en e-formation. Traité E-Form. Adultes. 263–286 (2019)
[18] A. Pereira, et al.: Universidade Aberta's pedagogical model for distance education: a university for the future. Universidade Aberta, Lisbon, Portugal (2008)
[19] D. Pedrosa, et al.: Self-regulated Learning in Computer Programming: Strategies Students Adopted During an Assignment. In: Immersive Learning Research Network. pp. 87–101. Springer International Publishing, Cham (2016)
[20] M. Chaves, Abordagem SimProgramming: e-learning no ensino de programação em ambiente virtual, http://hdl.handle.net/10400.2/7930, (2017)
[21] M.C. English, A. Kitsantas, Supporting Student Self-Regulated Learning in Problem- and Project-Based Learning. Interdiscip. J. Probl.-Based Learn. 7, (2013). https://doi.org/10.7771/1541-5015.1339
[22] J.R. Savery, Overview of Problem-based Learning: Definitions and Distinctions. Interdiscip. J. Probl.-Based Learn. 1, (2006). https://doi.org/10.7771/1541-5015.1002
[23] A. Pereira, et al.: Modelo pedagógico virtual da Universidade Aberta: para uma universidade do futuro. Universidade Aberta (2007)
[24] D. Pedrosa, et al.: Co-regulated Learning in Computer Programming: Students Co-reflection About Learning Strategies Adopted During an Assignment. In: Technology and Innovation in Learning, Teaching and Education. pp. 13–28. Springer International Publishing, Cham (2019)
[25] C.W. Tsai, Applying web-based co-regulated learning to develop students' learning and involvement in a blended computing course. Interact. Learn. Environ. 23, 344–355 (2015). https://doi.org/10.1080/10494820.2013.764323