Impact of Auto-evaluation Tests as Part of the Continuous Evaluation in Programming Courses

C. Rubio-Escudero[1], G. Asencio-Cortés[2],
F. Martínez-Álvarez[2], A. Troncoso[2], and J. C. Riquelme[1]

[1] Department of Computer Science, University of Seville, Spain
crubioescudero@us.es,riquelme@us.es
[2] Department of Computer Science, University Pablo de Olavide, ES-41013 Seville, Spain
guaasecor@upo.es,fmaralv@upo.es,atrolor@upo.es

Abstract. The continuous evaluation allows for the assessment of the progressive assimilation of concepts and the competences that must be achieved in a course. There are several ways to implement such continuous evaluation system. We propose auto-evaluation tests as a valuable tool for the student to judge his level of knowledge. Furthermore, these tests are also used as a small part of the continuous evaluation process, encouraging students to learn the concepts seen in the course, as they have the feeling that the time dedicated to this study will have an assured reward, binge able to answer correctly the questions in the continuous evaluation exams. New technologies are a great aid to improve the auto-evaluation experience both for the students and the teachers. In this research work we have compared the results obtained in courses where auto-evaluation tests were provided against courses where they were not provided, showing how the tests improve a set of quality metrics in the results of the course.

Keywords: Auto-evaluation, Continuous Evaluation, Test Questions, New Technologies

1 Introduction and Background

In the Education system, the evaluation is the way to know the degree of learning achieved by the student. Traditionally, the evaluation has focused on the final stage of learning and, in general, the student organized his learning according to the type of evaluation followed. As a consequence of the process of convergence towards the European Higher Education Area [4], the evaluation acquires a new dimension by focusing the learning process on the student [5]. In this sense, it must be correctly designed to allow assessing whether the student has reached, as a goal, not only the knowledge but also the competences previously defined by the teacher for a specific course. Therefore, the continuous evaluation appears as an appropriate tool that assesses the progressive assimilation and development of the contents of the course and of the competences that must be achieved. In

this way the teacher can carry out a greater and better follow-up of the progress in the student's learning.

There is a wide range of activities that can be carried out for the continuous evaluation process, such as resolution of practical scenarios, questions to develop theoretical concepts, multiple selection tests, true or false propositions, planning of debates on current issues, critical comment, preparing a topic at home using online resources, oral presentations of topics, preparation of tables and comparative diagrams, fill the gap type exercises, among others. In this work, we focus on the continuous evaluation aided by auto-evaluation test questions, that are provided along with the solutions to the students in the class. There are several publications stating that students show significantly more favorable attitudes towards multiple choice test format compared to essay type formats [11, 12].

The continuous evaluation system proposed is based on the use of new technologies, which represent a qualitative step in the learning process for both students and teachers [1, 7]. The technology platforms used for the auto-evaluation tests are Blackboard Learn [3], a virtual learning environment and course management system and a blog developed specifically for the programming courses (http://laprogramacionnoesunarte.blogspot.com.es/). Blogs and other Web 2.0 applications have shown to be very successful in aiding the learning process on different areas [2, 10].

Our proposal is to provide the students with auto-evalution test questions throughout the course's development, and to use some of those questions as a small part of the continuous evaluation system itself, thus encouraging students to revise their knowledge on the course concepts based on the results of the auto-evaluation. This methodology has been applied in programming courses of freshmen both in the Computer Engineering and Health Engineering degrees at University of Seville, Spain (US). We compare the results obtained by the students on these courses in terms of different quality metrics to other programming courses of freshmen in the University of Pablo de Olavide of Seville, Spain (UPO), where no auto-evaluation tests have been provided to the students. We will show that using auto-evaluation tests improves the course results related to number of students attending the continuous evaluation and number of students attending the final exam if they did not pass by continuous evaluation.

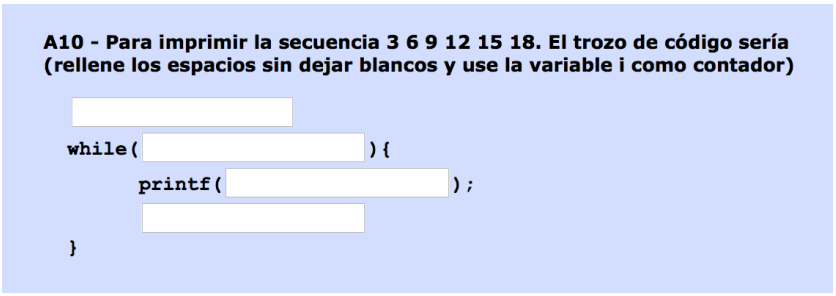
The rest of the article is organized as follows. Section 2 describes in detail how the auto-evaluation tests have been created and provided to the students. Section 3 shows the comparison of the results obtained by the two groups of freshmen compared: US with auto-evaluation tests vs. UPO without auto-evaluation tests. In Section 4 we highlight the main findings of this job.

2 Methodology

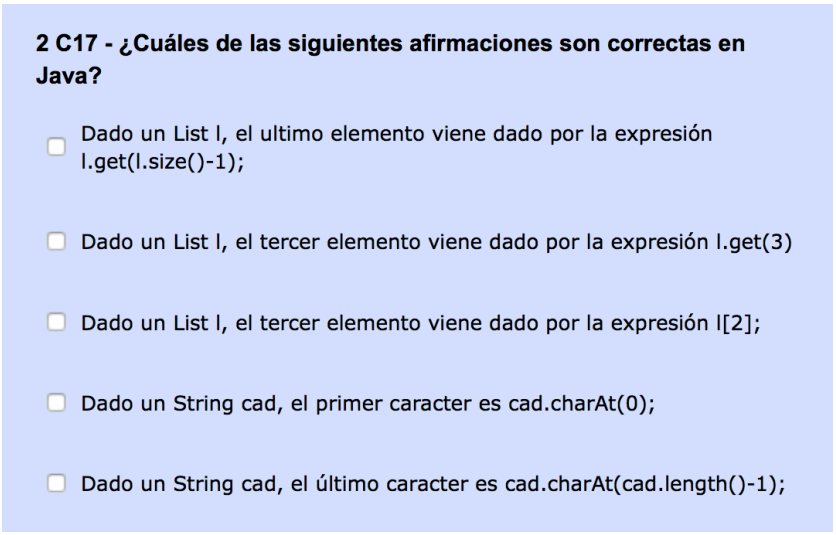
In this section we describe the method used to create the auto-evaluation tests, which are taken into account as a small part of the continuous evaluation system, thus encouraging students to revise their knowledge on the course concepts based

on the results of the auto-evaluation. The details of this technique are now presented.

For each of the conceptual blocks of the course, a set of test questions are provided, usually between 20 and 30. The questions include theoretical concepts as well as practical cases. The students gain access to this tests when the block concepts have been explained in class, and they can practice with the questions throughout the rest of the course. The questions follow two formats: fill the gaps (See Fig.1(a)) or multiple choice (See Fig.1(b)) [8, 9]. The questions can be accessed as many times as the student needs, and the results for each test answered will be provided along with the correct answers for each question.



(a) Fill the gaps question



(b) Multiple choice question

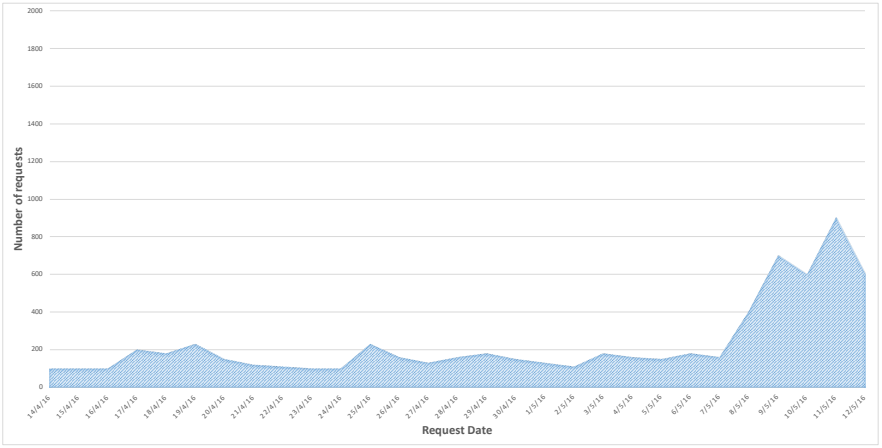
Fig. 1: Examples of auto-evaluation test questions (screen capture from the blog in the original language).

The continuous evaluation method[6] is made up of several parts. There are theoretical and practical sections, with the theoretical part being 60% of the final mark and the practical being the 40%. The theoretical part consists in 4 exams throughout the course. Each exam has 2 parts: the first one, some test questions very similar to the ones in the auto-evaluation tests, the second one writing some code to solve some problem. The test questions represent 25% of the exam's mark. The practical part consists in an exam taken in the computer lab, where the students have to solve a given problem

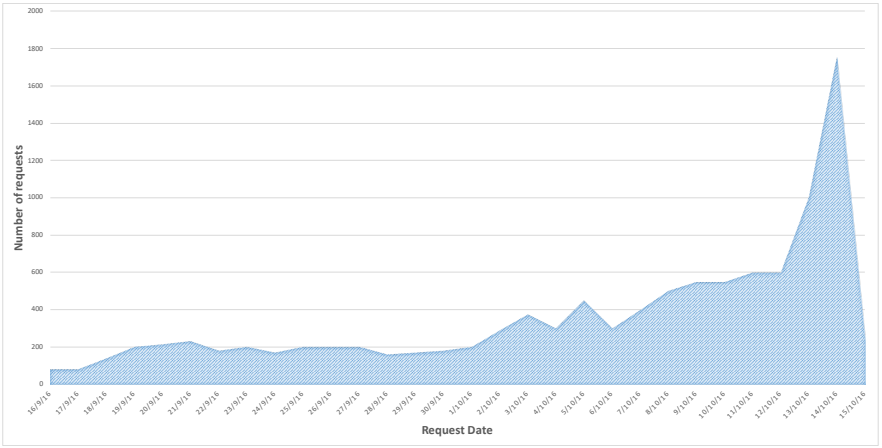
Therefore, a small portion of the final mark is obtained with the test questions provided for each block of the course are asked to the students. In particular, 15% of the final mark. This fact, knowing some of the questions that will take part in the continuous evaluation system, encourages the students to learn the concepts seen in the course, as they have the feeling that the time dedicated to this study will have an assured reward, as they will be able to answer correctly the questions in the tests. In Fig. 2 we can see some graphic representation of the number of times the tests have been accessed.Tests access dramatically increases the days previous to an evaluation. As the test questions in the exams are only an small part of the continuous evaluation system, 15%, there is no risk of the students passing the course only studying the auto-evaluation test questions.

Regarding the platform in which the test questions are developed, we initially used the Blackboard Learn software [3], as it is the virtual platform for our course and it provides a mechanism for this kind of task. Blackboard Learn is a virtual learning environment and course management system developed by Blackboard Inc. It is Web-based server software which features course management, customizable open architecture, and scalable design that allows integration with student information systems and authentication protocols. Its main purposes are to add on-line elements to courses traditionally delivered face-to-face and to develop completely on-line courses with few or no face-to-face meetings. We found it suitable to create a database of test questions for each block that could be reused and updated from one academic course to the next.

However, we found it difficult to extract statistics about the platform use from Blackboard Learn and therefore Dr. Riquelme developed a blog in which the tests along with other information for the course are stored (theoretical descriptions, exercises, exams from past years, media related content), so that we can more easily access the information related to the tests. To create the auto-evaluation test questions inside the blog, we have developed a specific tool written in Java to produce Dynamic HTML (CSS3, JavaScript and HTML5) from Microsoft Word documents which include questions and answers. In first place, all questions for the studied subjects were written in Microsoft Word documents. Then, our tool was executed producing the web pages in HTML. Finally, the page source code was embedded inside the blog.



(a) Access to tests for PF



(b) Access to tests for OOP

Fig. 2: Access to tests on the days previous to an exam. Two subjects were analyzed: Programming Fundamentals (PF) and Object Oriented Programming (OOP) (see section 3).

3 Results

Once the auto-evaluation tests were defined and planned properly through the continuous evaluation system of the studied course, a set of quality measures were assessed in order to assess the impact of such auto-evaluation tests into the student learning process.

For such purpose, two subjects were used: Programming Fundamentals (PF) and Object Oriented Programming (OOP). Both belong to the first course of the Computer Engineering degree in the US and UPO Universities. Whereas PF

is based on the structured programming paradigm using the C language, OOP is based on the object oriented paradigm using the Java language.

To compare the effect of introducing the auto-evaluation tests, two groups of freshmen from different universities were analyzed. Although the studied courses belong to different universities, their contents are very similar and suitable to be compared. Two academic years were analyzed: 2014/15 and 2015/16.

A set of evaluation metrics were assessed for each course, academic year and university. Specifically, the following five metrics were computed: Enrolled, CE attendance, CE passing students, CE failed & final exam and Only final exam.

The metric Enrolled counts the number of students enrolled in the course. The metric CE attendance refers to the number of students who attended the exams of the continuous evaluation. The CE passing metric means the number of students who passed the course by continuous evaluation. The metric CE failed & final exam counts the students who failed the continuous evaluation and attended final ordinary exam. Finally, the metric named Only final exam is referred to the number of students who did not follow the continuous evaluation and only came to the final ordinary exam.

From the metrics explained above, four indexes were derived: CE Attendance/ Enrolled, CE Passing/Attendance, CE Passing/Enrolled and CE failed†.

The index CE Attendance/Enrolled is the ratio between the number of students who attended the exams of the continuous evaluation and the number of students enrolled in the course. The index CE Passing/Attendance refers to the ratio between the number of students who passed the exams of the continuous evaluation and the students who attended the exams of the continuous evaluation. The CE Passing/Enrolled index means the ratio between the number of students who passed the exams of the continuous evaluation and the number of students enrolled in the course. Finally, the index named CE Failed† is the number of students who failed the continuous evaluation and did not come to the ordinary final exam divided by the number of students who failed the continuous evaluation.

Tables 1 and 2 show the results achieved for each course and course in terms of the metrics and indexes explained before. Specifically, Table 1 shows the values obtained for the course OOP, while Table 2 shows the results achieved for the course PF. Both tables include two subcolumns named *Yes* and *No* that indicate if the autoeevaluation tests were applied or not. Results with autoeevaluation tests were obtained in the US while those without them were obtained in the UPO. The higher values of indexes were highlighted using text in bold within tables.

As it can be seen from the results shown in Tables 1 and 2, the number of enrolled students were higher in the US than int the UPO. For such reason, the features to be compared in this study were the indexes computed (CE Attendance/Enrolled, CE Passing/Attendance, CE Passing/Enrolled and CE Failed†) rather than the metrics, because the former are relative values.

On the one hand, according to results achieved, we see that the ratio CE Attendance/Enrolled is significantly higher when auto-evaluation tests were ap-

OOP	2014/15		2015/16	
	Yes	No	Yes	No
Enrolled	131	83	131	81
CE attendance	81	37	71	40
CE passing	25	9	40	17
CE failed & final exam	37	11	16	6
Only final exam	10	1	2	3
CE Attendance/Enrolled	**61.83%**	44.58%	**54.20%**	49.38%
CE Passing/Attendance	**30.86%**	24.32%	**56.34%**	42.50%
CE Passing/Enrolled	**19.08%**	10.84%	**30.53%**	20.99%
CE Failed†	33.93%	**60.71%**	48.39%	**73.91%**

Table 1: Evaluation metrics and indexes achieved in the course OOP. Columns named *Yes* and *No* indicate whether the auto-evaluation tests were applied.

PF	2014/15		2015/16	
	Yes	No	Yes	No
Enrolled	112	66	123	73
CE attendance	54	16	95	30
CE passing	23	12	29	11
CE failed & final exam	22	2	28	8
Only final exam	7	5	0	9
CE Attendance/Enrolled	**48.21%**	24.24%	**77.24%**	41.10%
CE Passing/Attendance	42.59%	**75.00%**	30.53%	**36.67%**
CE Passing/Enrolled	**20.54%**	18.18%	**23.58%**	15.07%
CE Failed†	29.03%	**50.00%**	57.58%	**57.89%**

Table 2: Evaluation metrics and indexes achieved in the course PF. Columns named *Yes* and *No* indicate whether the auto-evaluation tests were applied.

plied. Specially, for the subject PF, the attendance to the continuous evaluation in presence of auto-evaluation tests is close to the double of those without them (48.21% vs 24.24% and 77.24% vs 41.10%). Moreover, the ratio CE Passing/Enrolled is also higher when auto-evaluation tests were applied.

On the other hand, the ratio CE Passing/Attendance varies depending on the course. While such ratio was higher with auto-evaluation tests for OOP, it was lower for PF. This can be due to the specific characteristics of the language paradigm. Specifically, auto-evaluation questions in OOP could share more related concepts to the exam questions than PF. Note that the object oriented

programming has more language overhead (classes, interfaces, attributes, methods, heritage, polymorphism, among others) than the structured paradigm (variables, functions, among others). As a consequence, OOP has less algorithmic load and the problems solved are simpler than those addressed in PF. Therefore, we conclude that the auto-evaluation questions in the course PF could be improved including more algorithmic load in order to increase the ratio CE Passing/Attendance.

It is desirable that the students who attend the continuous evaluation and fail it attend the final ordinary exam, as a sign of favorable learning during their continuous evaluation. In this sense, the ratio CE Failed† measures the drop-out rate after the continuous evaluation. As it can be seen in the results, the ratio CE Failed† was considerably lower when the auto-evaluation tests were applied (33.93% vs 60.71% and 48.39% vs 73.91% for OOP, 29.03% vs 50.00% for PF).

4 Conclusions

This paper evaluates the impact of using tests as part of the continuous evaluation in Programming courses. Students have been provided with auto-evaluation questions. Some of these tests have taken part of the continuous evaluation system itself, thus encouraging students to revise their knowledge on the course concepts based on the results of the auto-evaluation. This methodology has been applied in programming courses of freshmen both in the Computer Engineering and Health Engineering degrees at University of Seville, Spain (Java and C language programming). Results obtained by the students on these courses in terms of different quality metrics have been compared to other similar subjects at Pablo de Olavide University of Seville, Spain, where no auto-evaluation tests have been provided to the students. It has been shown that the use of such tests enhance the students performance both in terms of attendance and passing exams.

Acknowledgments.

The authors want to thank the financial support given by the Spanish Ministry of Economy and Competitivity project TIN2017-88209-C2-R. Also, this analysis has been conducted under the Innovation Teaching Project helps (2015 and 2016) by the University of Seville and Pablo de Olavide University.

References

1. D. Acemoglu. Why do new technologies complement skills? directed technical change and wage inequality. *The Quarterly Journal of Economics*, 113(4):1055–1089, 1998.
2. M. N. K. Boulos, I. Maramba, and S. Wheeler. Wikis, blogs and podcasts: a new generation of web-based tools for virtual collaborative clinical practice and education. *BMC Medical Education*, 6(1):41, 2006.

3. P. Bradford, M. Porciello, N. Balkon, and D. Backus. The blackboard learning system: The be all and end all in educational instruction? *Journal of Educational Technology Systems*, 35(3):301–314, 2007.
4. Bologna Declaration. The european higher education area. *Joint Declaration of the European Ministers of Education*, 19, 1999.
5. A. M. Delgado García and R. Oliver-Cuello. La evaluación continua en un nuevo escenario docente. *RUSC. Universities and Knowledge Society Journal*, 3(1), 2006.
6. S. L. Deno, D. Marston, and P. Mirkin. Valid measurement procedures for continuous evaluation. *Exceptional Children*, 48(4):368–376, 1982.
7. D. Kellner. New technologies/new literacies: Reconstructing education for the new millennium. *Teaching Education*, 11(3):245–265, 2000.
8. D. Nicol. E-assessment by design: using multiple-choice tests to good effect. *Journal of Further and Higher Education*, 31(1):53–64, 2007.
9. L. Prodromou. *The backwash effect: from testing to teaching*. Oxford University Press, 1995.
10. N. Saeed, Y. Yang, and S. Sinnappan. Emerging web technologies in higher education: A case of incorporating blogs, podcasts and social bookmarks in a web programming course based on students' learning styles and technology preferences. *Educational Technology & Society*, 12(4):98–109, 2009.
11. D. Tozoglu, M. D. Tozoglu, A. Gurses, and C. Dogar. The students' perceptions: essay versus multiple-choice type exams. *Journal of Baltic Science Education*, 6, 2004.
12. M. Zeidner. Essay versus multiple-choice type classroom exams: the student's perspective. *The Journal of Educational Research*, 80(6):352–358, 1987.