A cost–benefit analysis of continuous assessment

Author 1, Author 2, Author 3, Author 4 and Author 5

Keywords: Continuous assessment, Cost-benefit analysis, CS1

Abstract: The first course on programming is fundamental in the School of Informatics. After a major redesign of the Programming-1 course in 2006 to give it a more practical flavor, an increasing number of measures have been undertaken over the years to try to increase its pass rate while maintaining a fixed quality level. These measures, that can be roughly summarized as *an important increase in assessment*, imply an increase in the workload of both students and instructors that does not correspond to the increase of pass rate they provide. In this paper, and within the context of this course, we analyze quantitatively the amount of work required from faculty to implement the series of measures and we conclude that, within this course, continuous assessment is expensive and has reached its limit.

1 INTRODUCTION

The Programming-1 course at the School of Informatics of the University of Somewhere involves about 450 first-year students and about 15 faculty members and two coordinators per semester. In September 2006, the course's coordinators redesigned the course adopting a "learn-by-doing" approach: from the very beginning, students were expected to solve a strategically organized collection of programming problems. An integral part of the course was an online *programming judge* that automatically verifies in real time whether student solutions are correct. Students are organized into groups that receive 3 lecture hours and 3 lab hours a week. An account of the first two years of this experience is given in (Giménez et al., 2009), where it is shown that, unfortunately, a high number of students failed to pass the course. Indeed, the data collected on the online judge showed that the effort that most students dedicated to the course was far from the workload required (7.5 ECTS[1]). In fact, the data compiled by the system and the observations made by instructors showed also that, in general, most students did not even invest in the course the time needed for the theory lectures and laboratory sessions.

Deeply concerned and committed to the challenge of helping students to achieve the practice and knowledge required to attain a passing mark, the Programming-1 academic staff have been introducing, over time, a series of measures with the intention of incentivize students to work harder, more autonomously and more continuously while maintaining the general goals, level and approach of the course. As a consequence, the course has suffered several amendments, which, overall accounts for an important increase of continuous assessment of students at the expense of a parallel increase in the workload of the faculty.

As lecturers in a Technical University we are deeply engaged in the development of a learning society. However we found a big gap between general theories (Arrow, 1962; Solow, 1997; Stiglitz and Greenwald, 2014) and our everyday lecturing task. The time devoted to teaching is a limited resource and it should be optimized with no detriment of its quality. To do so, a fundamental issue is to estimate the cost-benefit of the different faculty tasks. This paper aims to be a proposal on this direction and can provide a starting point for fruitful discussions.

We perform a cost–benefit analysis that determines the impact of those continuous assessment's measures by contrasting the pass rate of the students with the workload of the instructors. To do so, we describe the series of measures that have been applied to the course and we present some statistics about their implementation in the later courses. We propose a simple way to interpret them under economic terms by means of *productivity* and *marginal gain* notions (Varian, 2005).

At first, our methodology consists in recognizing which changes have been applied to the course since its kick-off. This results in a series of measures labeled with time-stamps that serve as the time basis of our study. Then, following a long tradition in edu-

[1]European Credit Transfer and Accumulation System (ECTS) is a standard for comparing the study attainment of higher education across the European Union. One ECTS credit corresponds to 25 hours.

cation analysis (Bowles, 1970) (Hanushek, 2008), we use the rate of students passing the course as our primary measure of production. Such a temporal evolution of evaluative activities is a well established subject (Martín-Carrasco et al., 2014). Specifically, we use the following magnitudes:

- N_t denotes the total number of students at time t,
- P_t denotes the number of students passing the course at time t, therefore
- $100P_t/N_t$ corresponds to the *pass rate* at time t and,
- W_t denotes the total number of working hours required from the faculty members to teach the course at time t.

It is clear that our model and our statistical data is limited and does not take into account several pedagogical, psychological and sociological aspects that affect both the behavior of students and faculty members. Nevertheless, we think that it can provide insights in the way this massive course has evolved as well as tools for future directions.

According to the results reported in this paper, the benefit of incrementing the load of continuous assessment reaches soon a limit, regarding the rate of success of the students and the instructors' workload.

The forthcoming sections are organized as follows. First, in Section 2, we give an overview of the context and original design of the course, as well as a description of the different evaluative activities proposed over the later years. Then, we describe the impact of those actions on the pass rate of the students in Section 3. The total workload induced by a course of this nature is described in Section 4. An analysis in economical terms is carried out in Section 5. Finally, in Section 6, we present our concluding remarks.

2 THE COURSE: CONTEXT, DESIGN AND EVOLUTION

Context. In the Spanish educational system, the admission to Universities (in terms of number of vacancies, threshold qualification, etc.) is established by a public government office which is independent of those Universities. After concluding their secondary studies, students must do a several multi-subject general tests in order to be able to apply for a University vacancy. Those tests are countrywide. Thus, in general, first-year students are not previously filtered by any specific admission exam designed by the Universities. In order to compensate that lack of specific filtering, the first year in most of the degrees becomes somehow a selective procedure. That happens also at the School of Informatics where, at the end of the first year, many of the approx. 450 incoming students will drop out. The current selection criteria at the School of Informatics is that, all four subjects composing the first year of the degree must be successfully passed in at most four consecutive semesters. To illustrate this, Table 1 shows the percentage of students passing the first four semesters at their first try in the School of Informatics.

Subject	#Stu	%Enr	%Exa
Algebra	809	24%	32%
Computers-1	744	44%	57%
Physics	751	39%	48%
Programming-1	721	20%	32%

Table 1: For each subject in the academic year 2006-2007: #Stu is the total number of students; %Enr is the percentage of students who pass; and %Exa is the percentage of students who pass among those who took the final exam.

We are not entering in the debate of what is an acceptable pass rate. However, there is a general agreement that these percentages are too low and can be improved. Nevertheless, this is said under the agreement that the minimum level of required programming skills are correctly designed and should not be changed.

With respect to Programming-1, by 2006 there was a big consensus on the fact that, with independence of their grades, most students did not master the programming skills needed for subsequent courses. Consequently, the Programming-1 course was completely redesigned in September 2006. In the following subsection we give a short overview of the new course (full details can be found in (Giménez et al., 2009)).

Design. The main goal of the new Programming-1 course was to ensure that students would learn and master basic practical programming skills. In order to achieve this goal, the course was organized around the notion of "programming problems", that is, small programming tasks, specified in terms of valid inputs and desired outputs, for which students must write a small, correct and efficient `C++` program performing the required tasks. During the course, students would solve as many programming problems as possible among a collection containing more than 300 problems. The collection was conveniently organized by topic and level. Some of these problems were expected to be solved individually during the laboratory sessions with the occasional help of an instructor; some others were expected to be solved without the

instructor's immediate support. Theory sessions were used, as usual, to introduce the techniques and tools required to solve the problems. In the exams, students were asked to solve programming problems with a difficulty similar to those in the collection. Those exams took place in the laboratory room where students where used to work every week.

In order to apply this methodology, an online educational programming judge was developed. Online programming judges are web systems that store a repository of problems with the facility to check whether a candidate solution is correct. The judge executes the submitted program on a set of public and private test cases, and matches the obtained outputs with the expected ones. Online judges originated in programming contests such as the UVa Online Judge (Revilla et al., 2008), and have been widely adapted to educative settings (Ihantola et al., 2010; Verdú et al., 2012; Tonin et al., 2012). In particular, the judge of Programming-1 has evolved into www.jutge.org, an open access virtual learning environment for computer programming (Petit et al., 2012); see Figure 1.

The judge was an inflection point for our programming courses. Indeed, this kind of *public good* offers advantages to the students since it is a tool to freely work and study as it may be used 24/7. Also, it is an invaluable help to instructors to track the work and evolution of their students and for the assessment of exams.

In particular, the judge was also used during the exams, where it was compulsory to submit a correct solution in order to be evaluated. After the exam, only the correct solutions were additionally checked by human instructors, mainly to grade their adequacy to general quality criteria. The aim of this strict rule was to force students to put a lot of effort to practice on their own. However, our students perceived it as really unfair.

The results of introducing this way of teaching and learning to program were not as successful as expected at the very beginning. In consequence several measures were taken with the intention of turn the situation around. In what follows, we describe those measures and comment on and how they affected the evolution of the course.

Evolution. In order to try to improve its learnability, and therefore its pass rate, a series of gradual modifications were applied to the course as initially conceived. As it will be seen in next section, none of these measures was able to boost the rate of success on its own, a fact that resulted in their continued application over the years.

Specifically, this is the list of the main measures

Figure 1: Jutge.org with two of its verdict icons: The green light icon for submissions that pass all the test cases of a problem, and the red light icon marking submissions that fail some test case.

applied to the course organization since its kick-off up today, dividing the total time span into eight periods $t_0, \ldots, t_7$:

t_0 **Kick-off** (2006-2007): The first edition of this course started with 3 hours of theory lectures addressed to groups of 60 students and 3 hours of practical (a.k.a., laboratory) sessions addressed to groups of 20 students. There were two exams (a mid-term exam and a final exam) consisting of two practical problems each. The exams took place at the same rooms where students were used to work every week on their practical lessons. The students were asked to solve the problems, to implement their solutions and to submit their programs to the online judge. Each solution to a problems could be submitted more than once. Each submission gets a verdict from the online judge in a few seconds. Only those programs accepted by the Judge (labeled with a *green light*, see Figure 1) were then graded by the instructors. The rest (i.e., those labeled with a *red light*) were given the lowest mark: a zero, in the Spanish system.

t_1 **Introduction of quizzes** (2007-2008): In order to encourage the continuous work, four additional practical exams were distributed along the semester. Those exams consisted of an exercise of the same format and complexity as those solved in the practical sessions, and thus simpler than the problems included in the mid-term and final exam. The goal of this action was two-fold: first, to help students to get used to work under the same scenario as in the final exam, and second, to encourage them to work hard and get good marks by obtaining a percentage of the final qualification (10% at most) by working continuously and succeed those exams.

t_2 **Grading red lights** (2008-2009): Several lecturers and the majority of students considered that the fact of grading only those solutions labeled by the online judge as green was unfair. Therefore, the School of Informatics urged the people in charge of the course to eliminate this rule, and

to manually grade all the solutions (including the incorrect ones, the ones that did not succeed to get a green light by the online judge).

t_3 **Written final exam** (2009-2010): There was also the feeling among a few lecturers and some students that the fact that exams were only practical and ran in front of the computer, was also a cause of failure. In order to neutralize that opinion and minimize the effects of that situation, the old hands-on practical final exam was replaced by a final written (traditional) exam.

t_4 **New degree** (2010-2011): In September 2010 the School of Informatics introduced a new curriculum for the Degree of Computer Science to comply with the new law of the European Union regarding graduate studies. This new curriculum is the one that the Programming-1 course still currently follows. The most relevant changes were: (a) that the theory lectures were reduced from 3 to 2 hours, and (b) that practical exercises became mid-term exams with a greater weight on the final mark.

t_5 **Lists of problems to hand-in** (2011-2012): To enforce continuous work again, several lists of mandatory practical exercises for each topic of the course were introduced. Those lists were composed by normal exercises which were already in the set of problems composing the standard practical sessions of the course. Therefore, the students had the possibility to solve them before the exam. In order to be accepted in mid-term exams, students must properly solve and submit (via the online judge) around 70% of the exercises of those lists. The mid-term exams' problems were taken from these lists.

t_6 **Re-evaluation course** (2012-2013): As another effort to increase the rate of success, the School of Informatics introduced the concept of a *remedial exam* for students whose grade was not good enough but not bad enough. Once the usual course is finished, these students could apply for *remedial lectures* that consisted in intensive daily sessions of 2 hours each, during 6 consecutive days. Attendance to the lectures is mandatory. The re-evaluation course gives the students who are very close to pass the course, the right to a further examination. As in the normal course, that right is also conditioned to solving 70% of the problems of some proposed lists. If a student does not pass the re-evaluation exam, it keeps its original qualification of the course. Otherwise, he obtains the minimal mark that allows him to pass (a 5, in the Spanish system). No higher marks are to be obtained.

t_7 **Quizzes not from the lists** (2013-2014): In order to lead the students towards a more creative learning, the mid-term exams were designed as completely new problems. Those problems were unknown to the students by the time of the exam.

3 PASS RATE

Taking into account the history of the course detailed in the previous section, we now turn our attention to the evolution of the pass rate (or rate of success) as a function of the measures taken over time.

Figure 2 shows the pass rate from t_0 to t_7 (notice that t_0 is labeled as 2007 because the period of the course extends 2006-2007). The corresponding values are shown in Table 2. One can see that the proportion of students who pass the course in a first attempt started at around 20% and is now close to 40%. More specifically, it is also shown that:

- The introduction of quizzes at t_1 had almost no effect in the percentage of students who finished the course successfully.
- Grading red lights at t_2 had almost no effect on increasing the success of students (but, at least, it removed the feeling of unfairness).
- The introduction of a written final exam at t_3 modestly improved the percentage of success by a 3%.
- The adaptation to the new degree at t_4 had some effect, since this last modification boost the percentage of success to 30%, a real improvement, although still not good enough.
- Using lists of problems to hand-in for the quizzes at t_5 turned the percentage of success up to 41%. In spite of that positive result, the instructors were not pleased with this action since they got the impression that it reinforced memorizing programs rather than learning to program. The students concentrated too much on the problems of the lists and did not work progressively on problems of increasing difficulty. Without that background, they got often blocked and frustrated when tackling more difficult problems.
- The remedial exam action introduced at t_6 increased the rate of success from 41% to 50%. This amendment seems to suggest that there is an important percentage of students in the boundary of the passing mark who, with a bit more practice and personalized attention, succeed to pass the

Time	N_t	P_t	%
t_0-Kick-off	377	77	20%
t_1-Introduction of quizzes	492	102	21%
t_2-Grading red lights	497	105	21%
t_3-Written final exam	417	98	23%
t_4-New degree	493	145	29%
t_5-Lists of problems to hand-in	492	205	42%
t_6-Re-evaluation	465	232	49%
t_7-Quizzes not from the lists	436	166	38%

Table 2: Timestamp t, number of enrolled students (N_t), number of students who pass the course (P_t) and its percentage (%).

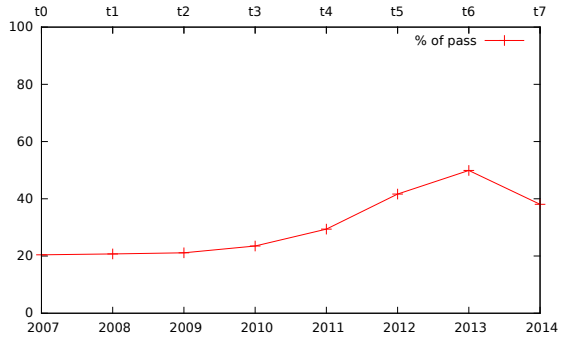

Figure 2: Percentage of students who pass the course by year (timestamp). A graphical representation of the last column of Table 2

Programming-1 course. The attendance to lectures also seems to have some influence on that success.

- Using problems that were not in the lists for exams at t_7, decreased the rate of success from 50% to 38%. This important decrement of the rate of success reinforces the risks mentioned for the initiatives taken at t_5. The impression of the faculty members is that this decrement is temporal and should improve in the forthcoming semesters.

4 WORKLOAD

Our goal in this section is to estimate the workload of the course in each of its stages ($W_t, t \in \{t_0, \ldots, t_7\}$), measured as the total number of working hours invested by the faculty members.

Computing W_t is difficult because every new edition of the course involves slightly different tasks to be done by faculty members with different profiles, dedications and efficiencies. Moreover, the faculty members involved in the course also changes from semester to semester. It is also the case, that the perception of each faculty member about the time he invested in each task is also different. We have approximated W_t by decomposing it into several tasks and conducting a survey among the current instructors of the course to get their time estimates for each of the tasks.

Therefore, W_t is conformed by:

- T_t: the number of working hours required to teach theory lectures.
- L_t: the number of working hours required to teach practical lectures.
- E_t: the number of working hours required to design, test and prepare exams.
- G_t: the number of working hours required to mark exams.
- V_t: the number of working hours required to supervise exams.
- C_t: the number of working hours required for coordinating the course.
- S_t: the number of working hours required for software maintenance.

Let us observe that, to a greater or lesser extent, all these quantities (except S_t) are dependent on the number of students. We have not included here, neither the working hours required for designing the course in the online judge, nor the working hours needed for designing the lists of problems to hand-in since those tasks are only performed by the course coordinators.

The values used here for estimating all these measures is the average over 14 answers received in the survey carried out at the end of the last semester (fall semester of the course 2014-2015). Since we did not have similar information from previous editions of the course (and it would have been almost impossible to obtain it), we extrapolated the results to past editions taking into account the way in which each applied amendment impacted the workload of each task. The technicalities for the calculation of the values of each task are given in Section 6, and the values are shown in Table 3.

Figure 3 shows the evolution over time of the workload of each of the tasks. One can see that the most significant weight contribution to the total cost W_t at each time step is due to practical lectures and theory lectures. In spite of that, the faculty members have the impression that the same does not apply to students. The tendency among many students as the course advances is not to attend those lectures.

One can also see that all the tasks —except V_t and G_t— are almost constant over time. Both V_t and G_t increase from t_0 to t_7 and are the principal reason why W_t also increases. Indeed, this behavior is as expected

Time	T_t	L_t	E_t	G_t	V_t	C_t	S_t	W_t	W_t/N_t
t_0-Kick-off	707	1697	24	91	76	34	300	2928	7.76
t_1-Introduction of quizzes	923	2214	36	178	99	40	300	3788	7.69
t_2-Grading red lights	932	2237	36	597	100	40	300	4241	8.53
t_3-Written final exam	782	1877	30	501	73	36	300	3598	8.62
t_4-New degree	617	2219	30	592	282	50	300	4087	8.29
t_5-Lists of problems to hand-in	615	2214	18	591	333	50	300	4120	8.37
t_6-Re-evaluation	602	2153	30	558	333	48	300	4022	8.64
t_7-Quizzes not from the lists	565	2022	30	524	314	47	300	3801	8.71

Table 3: Timestamp t, workload (in hours) of the tasks of the course.

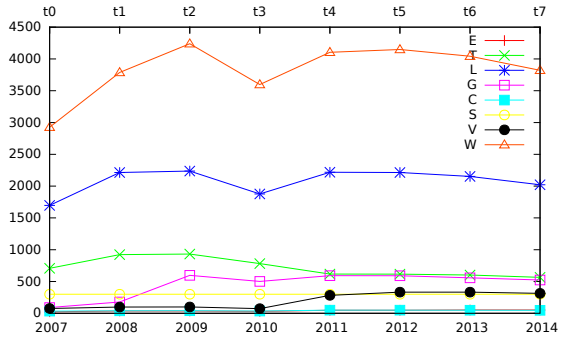

Figure 3: Workload (in hours) of each task of the course by year (timestamp). A graphical representation of the first eight columns in Table 3.

Figure 4: Hours of faculty work per student by year (timestamp). A graphical representation of the last column in Table 3.

because the amendments introduced along the years are mostly evaluative ones (i.e., directly or indirectly in the form of exams) and then it is natural that they mostly impact the tasks involved in designing, supervising and assessing exams.

Having the total workload hours W_t required for the course at every t, we can calculate the workload per student at time t as

$$W_t^e = W_t/N_t.$$

That measure indicates how many of the working hours of the faculty members are dedicated to each student or, in other words, what is the *cost* of every student in terms of working hours. Figure 4 shows the evolution of this cost over time. As the plot shows the cost per student increased by an hour from t_0 to t_7. As posted before, this extra hour exclusively corresponds to the increase in grading and supervising (G_t and V_t, respectively). Figure 4 shows that almost all the increment on the cost per student appears at t_2, when grading also the programs that obtained a red-light veredict by the online judge. In spite of that effort, we showed previously that the t_2 amendment did not have much influence on the pass rate.

5 ANALYSIS

In this section we go deeper into the analysis of the pass rate and the workload by relating them trough economic concepts.

On the surface, one can think that the whole evolution of the course (by means of the measures taken) is a great success since the whole workload (W_t) incremented by a modest 13% while duplicating the rate of success of the students. However, the measures taken did not affect the whole workload but only G_t and V_t that were (at least) triplicated. Therefore, the duplication of the rate of success does not seem to justify the triplication of $G_t + V_t$.

In order to get more insight on how each measure affects the pass rate, in the following we conduct a cost-benefit analysis relating workload and pass rate.

Economists define *productivity* or *effectiveness* as the ratio of outputs to inputs used in the production process (Varian, 2005). In our case, being very coarse and with all the safeguards and warnings required, one can see the number of students that succeed the course as the *output* and the total workload as the *input*. Therefore we talk about the *course productivity* Π_t as the ratio of these two quantities over time. This

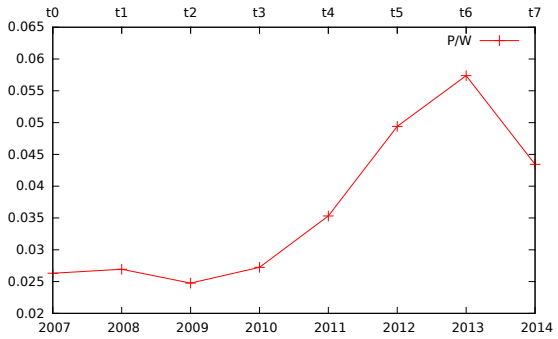

Figure 5: Productivity by year (timestamp).

is,

$$\Pi_t = P_t/W_t.$$

Figure 5 shows the behavior of Π_t from t_0 to t_7. Looking at the graph we can observe that Π_t has the typical *S*-curve shape in economic terms (Varian, 2005). The interpretation that it is generally given to this kind of curve is that if one reaches the inflection point (as it seems to be our case, although we have few observations after this inflection point) there is no sense to continue increasing the input (i.e., he workload of the course, in our case) because the increment has no impact on the output. In fact, it is negative.

Specifically the curve of Π_t has three sections that are noteworthy:

1. The first one is from t_1 to t_2 in which Π_t decreases due —as we already mentioned— to the grading of exam problems labeled with a red-light veredict by the online judge.
2. The second one is from t_2 to t_5 where it increases. During this period we can read from the curve that the course was being *productive* in the sense that the amendments applied were being effective as was the increase of the workload.
3. The last period corresponds to t_5 to t_7 in which Π_t decreases drastically. In this period the workload increased but the number of students that passed the course decreased. This is because the midterm practical exams are currently composed by new problems that are not known in advance to the students. Another fact that affects productivity in this period is the high amount of hours required

We consider now more closely the impact of the different measures over time. To capture the variation of work among periods we define

$$\Delta W_t = W_t/N_t - W_{t-1}/N_{t-1}$$

and for the variation of students that pass the course we define

$$\Delta P_t = P_t/N_t - P_{t-1}/N_{t-1}.$$

We compare both of them by the rate

$$\Delta_t = \Delta W_t/\Delta P_t.$$

Abusing again of economics terminology, we call this rate the *marginal gain* at time t of the undertaken measure (Varian, 2005).

Let us consider the following five general cases:

a) Case $\Delta W_t > 0$ and $\Delta P_t < 0$. Increasing the workload while decreasing the percentage of successful students corresponds to a very bad undertaken action. It seems definitely a situation to avoid.

b) Case $\Delta W_t < 0$ and $\Delta P_t < 0$ can be considered in general as a bad option; of course one wants to decrease ΔW_t but not with the consequence of decreasing also ΔP_t. However, if $|\Delta W_t| \gg |\Delta P_t|$ the undertaken action deserves to be carefully analyzed. It might be the case that a small decrease in ΔP_t is justified if it implies a huge decrease of the workload.

c) Case $\Delta W_t \geq 0$ and $\Delta P_t \geq 0$ and $\Delta W_t \gg \Delta P_t$. This corresponds to a big increase of work for a small increase in the number of passing students which is in general a situation to avoid.

d) Case $\Delta W_t \geq 0$ and $\Delta P_t \geq 0$ and $\Delta P_t \gg \Delta W_t$. This is a good case, a small increase in the quantity of work produces a big improvement.

e) Case $\Delta W_t < 0$ and $\Delta P_t > 0$. This is in general an outstanding measure. The larger the distance between ΔW_t and ΔP_t, the better the measure.

It is worth observing that the cases where Δ_t is really *unbalanced* deserve special attention. Such cases reflect an important disagreement between the effort (measured by ΔW_t) and the results (measured by ΔP_t).

Let us now interpret the amendments taken in this course over time in terms of the cases above. Table 4 shows the values of ΔP_t, ΔW_t and Δ_t along T.

t_1 **Introduction of quizzes** (2007-2008): This measure falls in Case e). Since $|\Delta W_t|$ is very small this was a moderately productive measure.

t_2 **Grading red lights** (2008-2009): This amendment falls in Case c). As we already mentioned this was a bad and unjustified measure that wastes a huge amount of resources. On one hand, it fails to take advantage of the online judge as a tool to help assessment, but on the other hand —and as a consequence— it requires working hours that could be probably invested in more productive activities.

t_3 **Written final exam** (2009-2010): The amendment falls in Case d). So it seems to be a good measure. Indeed, given that the red-light verdicts

Period	ΔP_t	ΔW_t	Δ_t
t_1-Introduction of quizzes	0.31	-0.07	-0.21
t_2-Grading red lights	0.40	0.83	2.11
t_3-Written final exam	2.37	0.10	0.04
t_4-New degree	5.91	-0.30	-0.05
t_5-Lists of problems to hand-in	12.25	0.11	0.01
t_6-Re-evaluation	8.22	0.26	0.03
t_7-Quizzes not from the lists	-11.82	0.07	-0.01

Table 4: Variations on rate of success, workload and marginal gain by timestamp.

have to be assessed, it is better to have written exams since the time to design them and supervise them is lower. However, this is only true in the context of marking red-light verdicts, but not in general. If compared against t_1 then it seems to be a worse measure.

t_4 **New degree** (2010-2011): This is a good measure that falls in Case e). Introducing mid-term exams with a significant weight over the final grade seems to have a very positive impact. The fact of decreasing the number of hours of theory lectures decreased the workload but did not seem to affect the rate of success.

t_5 **Lists of problems to hand-in** (2011-2012): This seems to be an outstanding amendment. It falls in Case e). However one has to be prudent with such kind of amendments. There is no doubt that it increased the rate of success while decreasing the workload, but the contents of mid-term exams were previously known by students. At the end, that might be a drawback because of the indirect use of mechanical learning, which is a risky practice. As G. Hardy strongly stated:

> *"It was an examination in which the questions were usually of considerable mechanical difficulty but unfortunately did not give the opportunity for the candidate to show mathematical imagination or insight or any quality that a creative mathematicians needs."* (Hardy, 1940)

t_6 **Re-evaluation** (2012-2013): This can be considered a good amendment, in spite of being very expensive and having a modest impact. It falls in Case d).

t_7 **Quizzes not from the lists** (2013-2014): This amendment falls in Case a). It seems to be a situation to avoid if one looks only into the numbers. However, when related to the situation at t_5, it seems to confirm our perception that the students are learning in a more mechanical way.

6 CONCLUSION

In this paper we describe the series of measures taken to increase the pass rate of the Programming-1 course since its inception. The successive introduction of these reported measures increased the weight of continuous assessment as a way to incentive students work.

However, it appears that despite taking all these measures, the rate of success of this massive course has not increased in a significant way, while in fact, all this increase of continuous assessment produced a significant increase of the faculty workload.

In order to obtain a quantitative assessment of the impact of each measure over both workload and pass rate we introduced two functions: productivity and marginal gain. These functions provided a way to rank the adopted measures between negative and positive. Our findings are valuable for the design of future strategies for this and similar courses.

The analysis tools reported on this study provide a way to analyze the effectiveness of new measures. According with their corresponding cost-benefit analysis, the measure can be maintained, tuned or withdrawn, under the general agreement that the current content of the course as well as the proficiency levels achieved by students who passed the course should not be changed.

In particular, for some of the adopted measures, it became clear that the amount of invested resources (faculty workload) did not justified their impact in the pass rate. Namely, the substantial overhead of grading red lights had almost no impact on passing rate. Other measures did have a positive impact without increasing the workload like the weights given to the different exams.

Some pedagogical strategies around the use of the online Judge as an automated aid to motivate, help and evaluate students that are used in this course have been successfully used also in other courses (such as Programming-2, Data structures and algorithms, Algorithms, Functional programming, among others). It could be interesting to extend this kind of analysis to

those courses.

We are aware that the scope of our study could be extended. We have focused uniquely on the passing rate but a finer analysis taking into account students marks and motivation might bring more insights in the effectiveness of every measure.

REFERENCES

Arrow, K. J. (1962). The economic implications of learning by doing. *The Review of Economic Studies*, 29(3):155–173.

Bowles, S. (1970). Towards an educational production function. In *Education, Income, and Human Capital*, pages 9–70. National Bureau of Economic Research.

Giménez, O., Petit, J., and Roura, S. (2009). Programació 1: A pure problem-oriented approach for a CS1 course. In Hermann, C., Lauer, T., Ottmann, T., and Welte, M., editors, *Proc. of the Informatics Education Europe IV (IEE-2009)*, pages 185–192, Freiburg. ISBN: 978-84-692-2758-9.

Hanushek, E. A. (2008). Education production functions. In *The New Palgrave Dictionary of Economics*. Palgrave Macmillan.

Hardy, G. (1940). *A Mathematicians Apology*. Cambridge. Reprinted with Foreword by C.P.Snow 1967. Current reedition, Cambridge Canto Edition, 2008.

Ihantola, P., Ahoniemi, T., Karavirta, V., and Seppälä, O. (2010). Review of recent systems for automatic assessment of programming assignments. In *Procs. of the 10th Koli Calling International Conference on Computing Education Research*, pages 86–93. ACM.

Martín-Carrasco, F. J., Granados, A., Santillan, D., and Mediero, L. (2014). Continuous assessment in civil engineering education - yes, but with some conditions. In *CSEDU 2014 - Proceedings of the 6th International Conference on Computer Supported Education, Volume 2, Barcelona, Spain, 1-3 April, 2014*, pages 103–109. SciTePress.

Petit, J., Giménez, O., and Roura, S. (2012). Jutge.org: an educational programming judge. In *Proceedings of the 43rd ACM technical symposium on Computer science education, SIGCSE 2012, Raleigh, NC, USA, February 29 - March 3, 2012*, pages 445–450.

Revilla, M., Manzoor, S., and Liu, R. (2008). Competitive learning in informatics: The UVa online judge experience. *Olympiads in Informatics*, 2:131–148.

Solow, R. M. (1997). *Learning from 'Learning by Doing' Lessons for Economic Growth*. Stanford University Press. Series: Kenneth J Arrow Lectures.

Stiglitz, J. E. and Greenwald, B. C. (2014). *Creating a Learning Society: A New Approach to Growth, Development, and Social Progress*. Columbia University Press. Series: Kenneth J Arrow Lectures.

Tonin, N., Zanin, F., and Bez, J. (2012). Enhancing traditional algorithms classes using URI online judge. In *2012 International Conference on e-Learning and e-Technologies in Education*, pages 110–113.

Varian, H. R. (2005). *Intermediate Microeconomics: A Modern Approach (7th Edition)*. W. W. Norton and Company.

Verdú, E., Regueras, L. M., Verdú, M. J., Leal, J. P., de Castro, J. P., and Queirós, R. (2012). A distributed system for learning programming on-line. *Computers & Education*, 58(1):1 – 10.

TECHNICALITIES

In this section we calculate the amount of working hours per task at each stage of our course based on the 14 responses that we obtained from current instructors and extrapolating these values to previous timestamps. Note that most measures, once taken, remain in force, thus the workloads involved are accumulated.

t_0 **Kick-off** (2006–2007): The course started with two kind of lectures, theory and lab, of 3 hours per week each. Theory lectures were given to groups of 60 students, and the survey says that in average it takes 1.5 hours to prepare one hour of theory lectures. This results in a total of 2.5 hours of work (preparation + lecturing). Since the course is 15 weeks long, we have:

$$T_{t_0}\ hours = (1+1.5)\frac{hours}{1h\ theory} \times 3\frac{1h\ theory}{week \times group} \times 15\ week \times \frac{N_{t_0}}{60}\ group \approx 1.7 \times N_{t_0}\ hour$$

where N_t is the total number of enrolled students at time t. Proceeding similarly for lab sessions and considering that the size of the lab groups is of 20 students, and that the preparation of each hour of lab sessions takes 1 hour, we have that $L_{t_0} = 2 \times 3 \times 15 \times \frac{N_{t_0}}{20}$.

There were two exams of 2 problems each. There were 2 turns of exams (morning and afternoon), all the students that have morning classes are examined with the same exam that is different from the exam of the afternoon students. So 4 problems should be prepared (2 per turn). Since each exam lasted for 2 hours and the students were distributed in lab rooms with 20 computers we have that $V_{t_0} = 2 \times 2 \times \frac{N_{t_0}}{20}$. We estimate that the preparation of each problem takes in average 3 hours of work (this include writing the statement, implementing the solution and designing the tests that the system requires to *judge* the submissions). Therefore, $E_{t_0} = 24$.

Only the solutions of students that obtained a green light for a problem were graded by hand,

and this was, approximately, a third of the students, so $G_{t_0} = 2 \times 2 \times 0.2 \times \frac{N_{t_0}}{3}$ hours, considering that grading one problem takes 12 minutes.

The coordination of the whole course has a fixed cost k plus a cost that depends on the number of students of each course that we estimate in one half hour per group of 10 students, then $C_{t_0} = k + 0.5 \times \frac{N_{t_0}}{10}$.

Finally, we are estimating that the software maintenance takes 4 hours a day yielding to $S_{t_0} = 4 \times 5 \times 15$ per course.

t_1 **Introduction of quizzes** (2007–2008): In this period 4 small mid-term exams were introduced in addition to the two original exams and they were applied also in two turns. Considering that the time required to prepare each small exam was 1.5 hours and that the time required to grade the small exam of one student was 6 minutes, this measure increased E_t and G_t to $E_{t_1} = E_{t_0} + 4 \times 2 \times 1.5$ and $G_{t1} = 4 \times 0.3 \times \frac{N_{t1}}{3}$. The workload of all the other tasks remained the same.

t_2 **Grading red lights** (2008–2009): When all the submissions (and not only the green labeled ones) have to be graded G_t was triplicated. $G_{t_2} = 4 \times 0.3 \times N_{t_2}$.

t_3 **Written final exam** (2009–2010): At this point the final exam was changed to be a written exam of 3 problems applied in only one turn to all the students (same exam for all students). The time to prepare a problem for a written exam is 2 hours (1 hour less than the time of a lab exam). Therefore $E_{t_3} = E_{t_1} - 2 \times 2 \times 3 + 3 \times 2$. The written exam lasts for 3 hours. Since larger rooms (with place for 60 students) can be used for a written exam, V_t decreased to $V_{t_3} = 2 \times \frac{N_{t_3}}{20} + 3\frac{N_{t_3}}{60}$.

t_4 **New degree** (2010–2011): With the new degree the hours of theory lectures per week decrease from 3 to 2 per group yielding $T_{t_4} = 2.5 \times 2 \times 15 \times \frac{N_{t_4}}{60}$. The evaluation system changed also. The big mid term exam disappeared. The four small mid term exams became formal exams of one problem each. Thus, $E_{t_4} = 4 \times 3 \times 2 + 2 \times 3$ The first 3 mid term lab exams lasts 1.5 hours each while the last one for 2.5 hours. The final exam lasts still for 3 hours, thus $V_{t_4} = (3 \times 2 \times 1.5 + 2 \times 2.5)\frac{N_{t4}}{20} + 3\frac{N_{t4}}{60}$.

t_5 **Lists of problems to hand-in** (2011–2012): A list of problems per exam to be delivered by the students before the exam was introduced. The problems of the exams were chosen from the problems of the list. The preparation of each lab problem decreased to 1.5 hours. Hence $E_{t_4} = 4 \times 2 \times 1.5 + 3 \times 2 = 18$. Finally, the preparation of the lists increase C_t by 10 hours, $C_{t_4} = k + 10 + 0.5\frac{N_{t_4}}{10}$.

t_6 **Re-evaluation** (2012-2013): This measure added 3 hours to E_t, 20 hours to T_t, 60 hours to L_t, 15 hours to G_t and 3 hours to V_t.

t_7 **Quizzes not from the lists** (2013-2014): This measure involved the creation of new problems for the mid-term lab exams, instead of taking them from the lists. This increased the time for preparing each problem from 1.5 to 3 hours. Thus $E_{t_5} = 4 \times 2 \times 3 + 3 \times 2 = 30$.