Continuous Assessment in the Evolution of a CS1 Course: the Pass Rate/Workload Ratio*

Maria J. Blesa, Amalia Duch, Joaquim Gabarró, Jordi Petit, and Maria Serna

ALBCOM Research Group. Computer Science Department
Universitat Politècnica de Catalunya - BarcelonaTech. Barcelona, Spain.
{mjblesa, duch, gabarro, jpetit, mjserna}@cs.upc.edu

Abstract. The first programming course (Programming-1, CS1) in the Informatics Engineering Degree of the Facultat d'Informàtica de Barcelona was completely redesigned in 2006 in order to reinforce the learn-by-doing methodology. Along the following eight years several pedagogical measures —mostly related with continous assessment— were introduced with the aim of increasing the pass rate of the course without lowering its high quality standards. This paper analyzes to what extent the added workload on faculty entailed by these measures affects the pass rate. We use a classical marginal cost-benefit approach —from Economics— to compare these two values along time. This process allows us to relate the evolution of the pass rate of students with the workload of the faculty through a productivity curve, as well as to assess the impact of each pedagogical measure. We conclude that, for this course, continuous assessment is expensive. In fact, abstracting from short term oscillations, the slope of the productivity curve is close to zero.

Keywords: CS1, marginal cost-benefit analysis, continuous assessment.

1 Introduction

The Programming-1 course (CS1) is the first course on programming taught in the Informatics Engineering degree at the Facultat d'Informàtica de Barcelona (FIB) of the Universitat Politècnica de Catalunya (UPC). It receives around 450 new students per academic year, and it requires about 15–20 faculty members and two coordinators. In September 2006, the course was completely redesigned adopting a "learn-by-doing" approach [1] that has been applied until now. The working basis is a strategically selected and carefully organized collection of programming exercises that must be solved using the C++ programming language. An integral part of the course is an online *programming judge* [8] that automatically verifies in real time whether a program proposed by a student is a

* Authors are partially supported by funds from the European Union (FEDER funds), from the Spanish Ministry for Economy and Competitiveness (MINECO) under grants TIN2013-46181-C2-1-R and TIN2012-37930-C02-02, and also from AGAUR of the Generalitat de Catalunya under grant SGR 2014:1137 (ALBCOM).

correct solution for a programming exercise. Such a tool is a very useful platform for supporting a programming course, since it is 24-hours available to the students and provides them with a large complementary training source for the self-organization of their own learning process. However, the first two years with the experience of using the programming judge showed that a high number of students failed to pass the course anyway [3]. According to the data collected in the online judge, most students invested far less time in the course than its required workload of the course (7.5 ECTS[1]).

As lecturers in a Technical University, we are deeply engaged in the development of a learning society. However, there is a big gap between general theories [1, 10, 11] and our everyday lecturing task. Concerned and committed to the challenge of helping students to achieve the theoretical and practical knowledge required to attain a passing mark, the CS1 academic staff has introduced a series of measures with the aim of motivating the 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 along time, which account for an important increase of continuous assessment for students at the expense of a simultaneous increase of the workload for the faculty members.

Since the time devoted to teaching is a limited (and even public, in our case) resource, it should be optimized with no detriment of its quality. To do so, it is fundamental to estimate the cost–benefit of the different faculty tasks involved in the course. Such information is definitely helpful for the assessment of the undertaken measures in terms of their productivity (or effectiveness). This paper is a proposal in this direction and it aims at providing a starting point for fruitful discussions and considerations.

The analysis of the temporal evolution of the evaluative activities of a course is a well established subject [7]. Following a long tradition in education analysis [2, 4], we use the rate of students passing the course as our primary measure of production. Specifically, we perform a cost–benefit analysis that determines the impact of the measures introduced in the last years in the CS1 course by contrasting the pass rate of the students with the workload of the instructors. We propose a simple way to interpret them under economic terms by means of *productivity* and *marginal gain* notions [13]. For that aim, we use the following basic 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 *workload* (as the number of working hours) that faculty members are required to invest on the course at time t.

It is clear that our model and our data are limited and do not take into account several pedagogical, psychological and sociological aspects that affect

[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 of student work.

the behavior of both 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 here, we can provide some criteria to assess the benefit of each of the introduced measures. We observe that the benefit of incrementing the load of continuous assessment reaches soon a limit, regarding the pass rate of the students and the workload of the instructors.

The forthcoming sections are organized as follows. First, in Sect. 2, we give an overview of the context in which the CS1 course is placed and we describe the original design of the course as in 2006. In Sect. 3, we detail the different measures on the evaluative activities proposed from 2006 until now. The impact of those measures in the pass rate of the students is analyzed in Sect. 4. Later, the total workload induced by the CS1 course is described and estimated in Sect. 5. Section 6 presents the cost-benefit and the marginal analysis method that will be used in Sect. 7 to carry out an analysis of the relation between pass rate and workload by means of economics concepts. Finally, Sect. 8 closes the article by presenting our concluding remarks.

2 Context and Design

It is usual around the world that, after concluding their secondary studies, students do some kind of multi-subject general exam in order to be able to apply for a vacancy at a University. In the Spanish educational system, those exams are valid countrywide and the later admission to Universities (in terms of number of vacancies, minimum grades, etc.) is established by an independent governmental office. Thus, except for a few very specific areas like art and sport, new incoming students are not previously filtered by any specific admission exam designed by the Universities where they end up studying. In order to compensate that lack of specific filtering, the first year in most of the undergraduate degrees becomes somehow a selective procedure. That is also the case for the FIB, where new students must succeed in (at least) the four subjects composing the first year of the degree in (at most) two years. Table 1 shows the percentage of students passing the first four semesters at their first try (i.e. in one year) at FIB in the academic year 2006-2007. We do not enter in the debate of what is an acceptable pass rate. However, the general impression was that the percentage for CS1 was too low and it should be improved.

By 2006, there was also a general consensus on the fact that most students of CS1 did not master fluently enough the programming skills needed for subsequent courses. Consequently, the course was completely redesigned in September 2006. This was done under the agreement that the level of required programming skills should be better designed and that its contents should not suffer changes in the forthcoming years. In the following subsection we give a short overview of the course (full details can be found in [3]).

The FIB offers CS1 twice a year, every academic year: once at the fall semester and once at te spring semester:

Table 1. Passing rate of each subject composing the first year in the Informatics Engineering degree in the academic year 2006-2007: %Enr is the percentage of students who pass over all the students enrolled; and %Exa is the percentage of students who pass over all who took the final exam.

Subject	%Enr	%Exa
Algebra	24	32
Computers-1	44	57
Physics	39	48
Programming-1	20	32

- The **Fall term** spans from September to February. It is during this term that new students arrive every year. For most of them, CS1 is the first serious attempt to learn to program. During this semester, the lecturers of CS1 put a special effort in motivating those new students for properly facing the Informatics Engineering degree.
- The **Spring term** spans from February to July. Virtually all students in this term are those who failed in the previous semester(s). Since they come from a negative experience, some students do not have a positive and constructive attitude in this second attempt. Contradictorily, the fact of having now some knowledge about the subject motivates a high level of absenteeism in the classes. In general, the students in this semester are harder to motivate.

In spite of the different nature of the students in each semester, the organization of the course does not change, nor its evaluation. Only the contents of the lessons (specially the practical ones) may be adapted according to the audience. In 2006, the course had 3 hours of theory lectures and 3 hours of practical lessons per week. In both semesters, the students were organized into groups: of 60–80 people for theory sessions, and of 15–20 people for practical classes. The theory sessions are, as expected, the place to introduce the concepts, techniques and tools required to acquire the necessary knowledge and to tackle the contents of the practical classes.

The main goal of CS1 was always to ensure that students learn and master basic programming skills. By 2006, there was the will to face that goal with a reinforcement of the practical side of the subject. Thus the course was organized around the notion of "programming problems", i.e., small programming exercises described with a clear statement in terms of valid inputs and desired outputs. The students must write a small, correct and efficient C++ program that solves the problem stated in the exercise and behaves as expected. During the course, students should solve as many programming problems as possible among the offered collection, which contains more than 300 problems.

The collection of programming exercises was conveniently designed and organized by topic and difficulty. Some of those problems were expected to be solved individually during the practical sessions with the help of an instructor. Some

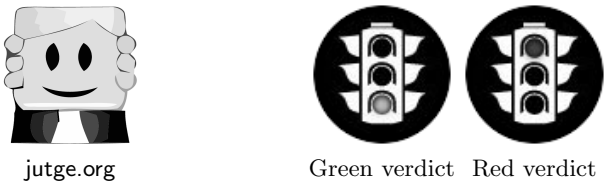

Fig. 1. jutge.org with two of its verdict icons: The green light icon for submissions that pass all the test cases of a problem, and the red light icon marking submissions that fail some test cases.

other problems were expected to be solved by the students on their own, without the instructor's immediate support and out of the regular lessons' time. During the exams, students were asked to solve programming problems with a difficulty similar to those in the course collection. Those exams took place in a laboratory room similar to the one where students used to work every week.

In order to apply this learn-by-doing methodology, an educational online programming judge was developed [8]. 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 [9], and have widely been adapted to educative settings [6, 14, 12]. In particular, the judge of CS1 has evolved into jutge.org, an open access virtual learning environment for computer programming [8] which is nowadays used for further programming courses in the Informatics Engineering degree and other degrees at UPC, as well as for training student teams for international programming competitions.

For every programming exercise in the judge, there is a clear statement with a description of the problem to solve. The statement includes a set of sample inputs and specifies the corresponding correct outputs. Once programmed, the student is expected to compile his program until a running code is produced. Then, the program should be tested for different inputs, those in the statement at first. The student must think, however, about other possible inputs not in that sample set. All these steps are performed in a raw environment since only a text editor and a C++ compiler is needed. Once the program is considered to be tested enough, the student submits it to the judge for its evaluation. At that point, the judge compiles the program again and tests it not only for the sample input set of the statement, but also for a private input set. After a few seconds the judge will come out with a verdict: a *green* light (green verdict) if the program is correct and passes all the input sets, and a *red* light (red verdict) if the program fails in some case (see Fig. 1). In this later case, a clue is feedbacked to the student to help her finding the mistake. Other verdicts are possible for describing other situations.

The use of the programming judge was an inflection point for our programming courses. Indeed, this kind of *public good* offers clear advantages to the students since it provides them with a huge source of exercises, gives them freedom at work and it may be used 24/7. It is also a valuable tool for instructors because it dynamizes the practical sessions and it allows to track the work and evolution of the students, as well as facilitating the organization and management of the exams.

The judge started to be used also for the exams, where it was compulsory to submit a correct solution in order to be evaluated. After the exam, the teachers only evaluated those programs with green verdict, mainly to grade their adequacy to general quality criteria. The aim of this strict rule was to make the students used to practice on their own, and to reinforce the goal of getting programs that run and not only algorithms on a paper. However, the students at that moment perceived it as an unfair method.

3 Evolution

The immediate results after introducing the methodology mentioned above were not as successful as expected. As a consequence, several measures were taken in the forthcoming editions with the intention of improving the situation. We now describe those measures and comment on how they have affected the evolution of the course. Unfortunately, we will see later that none of those measures was able to boost the pass rate on its own. We divide the total analysis time into eight periods $t_0, \ldots, t_8$, and sometimes refer to them as *timestamps*.

- t_0 **Kick-off** (2006-2007): The first edition of this course had 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 in their practical lessons. The students were asked to solve the problems, to implement their solutions and to submit their programs using the online judge. Each solution to a problem could be submitted more than once. Only those programs that obtained a green verdict were then graded by the instructors. All the other programs with a red verdict were given the lowest mark, i.e., a zero in the Spanish system.
- t_1 **Introduction of quizzes** (2007-2008): In order to encourage students to work in a more continued way, four additional practical exams were introduced along the semester. Those exams consisted of an exercise of the same format and complexity as those solved in the practical sessions, and thus generally simpler than the problems included in the mid-term and final exams. The goal of this amendment was two-fold: first, to help students to get used to work under the same scenario where the mid-term and final exams take place, and second, to encourage them to work hard and in a continuous way. They could obtain up to a 10% of the final qualification by succeeding in those exams. Still only green verdicts were graded by lecturers.
- t_2 **Grading red verdicts** (2008-2009): Some lecturers, and most of the students, considered that grading only those programs that obtained a green

verdict was somehow unfair. Therefore, the FIB urged the coordinators of the course to remove this restriction and force them to grade all the solutions manually, independently of its final verdict.

t_3 **Hand-written final exam** (2009-2010): The fact of only having computer-based exams was a significant novelty for students, because they were rather used to write down their exams on paper. Somehow, the feeling that computer-based exams were the reason for the bad results grew up among students. In order to neutralize that opinion and minimize the effects of that situation, the final exam changed back to a traditional hand-written format.

t_4 **New degree** (2010-2011): In September 2010, the FIB introduced a new curriculum for the Informatics Engineering degree to comply with the new European Union regulations on graduate studies. This new curriculum is the one that the CS1 course still follows nowadays. Two most relevant changes were introduced: (a) theory lectures were reduced from 3 to 2 hours per week, and (b) the continuous assessment increased in quantity and importance, by considering practical exercises as mid-term exams with a greater weight in the final mark than they had before.

t_5 **Lists of problems to hand-in** (2011-2012): In order to comply with the required increase of continuous assessment, lists of mandatory practical exercises for each topic of the course were introduced. Those lists were exclusively composed by exercises in the course collection of the online judge. Therefore, the students could work on them during their practical sessions and had the possibility to solve them before the exams. The exercises of the mid-term exams were taken from those lists. In order to have the right to participate in a mid-term exam, the students were required to achieve a green verdict in the online judge for at least 70% of the exercises of the corresponding lists.

t_6 **Re-evaluation course** (2012-2013): As another effort to increase the pass rate, the FIB introduced a second-chance exam for students who did not succeed the course, but whose final grade was close to the passing threshold mark (i.e., a five in the Spanish grading system). That re-evaluation course is a summarized 12-hour course that takes place once the usual course is finished. Attendance to the lectures is mandatory. As in the normal course, the right to get the remedial final exam is also conditioned to solving 70% of the problems of the proposed lists. If a student does not pass the exam, he keeps the original final qualification of the course. Otherwise, he gets a 5 as final mark and thus passes the course with the minimal qualification. No higher marks can be obtained.

t_7 **Mid-term exams with new exercises** (2013-2014): In order to lead the students towards a more creative and responsive learning process, the mid-term tests changed their composition to completely new problems that were unknown to the students by the time of the exam.

t_8 **Course diversification** (2014-2015): From 2006 to 2014, the collection of training exercises for CS1 in the online judge remained almost unchanged and only the evaluation of the course varied. Trying to further reinforce a creative learning, we devote an effort to set-up a diversified content for the course. Such a diversification might reinforce the idea of enrolling always

Table 2. For every timestamp t, number of enrolled students (N_t), number of students who pass the course (P_t) and its percentage (%), per semester and academic year.

	Fall			Spring			Year		
Timestamp	N_t	P_t	%	N_t	P_t	%	N_t	P_t	%
t_0-Kick-off	377	77	20	344	67	19	377	144	38
t_1-Introduction of quizzes	492	102	21	395	132	33	492	234	48
t_2-Grading red verdicts	497	105	21	380	136	36	497	241	49
t_3-Hand-written final exam	417	98	23	360	118	33	417	216	52
t_4-New degree	493	145	29	296	92	31	493	237	48
t_5-Lists of problems to hand-in	492	205	42	220	69	31	492	274	56
t_6-Re-evaluation course	465	232	49	181	83	46	465	315	68
t_7-Mid-term exams with new exercises	436	166	38	205	96	47	436	262	60
t_8-Course diversification	448	209	46	169	90	53	448	299	67

in a novel course. Further, it might motivate the students (specially, those repeating the course) to face the semester willingly.

4 Pass Rate

Taking into account the evolution of the course, as detailed in the previous section, we now turn our attention into the evolution of the pass rate as a function of the measures taken over time. For every timestamp, Table 2 shows the number of enrolled students and the number of students that pass the course, for every semester and academic year since 2006. From that information, Fig. 2 plots the pass rates. One can see that the proportion of students who succeed the course at a first attempt started at around 20% and is now slightly over 45%. Some additional observations can be done:

- The introduction of quizzes at t_1 had almost no effect in the percentage of students who finished the course successfully.
- Grading red verdicts at t_2 may have removed the feeling of suffering an unfair evaluation, but it had almost no effect on increasing the pass rate.
- The introduction of a written final exam at t_3 modestly improved the percentage of students who succeeded by a 3%.
- The adaptation to the new degree at t_4 seems to had some modest effect, since the percentage of the pass rate boosted to a successful 30%.
- Using lists of problems to hand-in for the quizzes at t_5 turned the percentage of students passing the course up to 41%. In spite of that positive result, the instructors were not completely pleased with this action since they got the impression that it motivated memorizing programs rather than learning to design and implement them. The students concentrated too much in the problems of the lists and did not use the other problems to work more

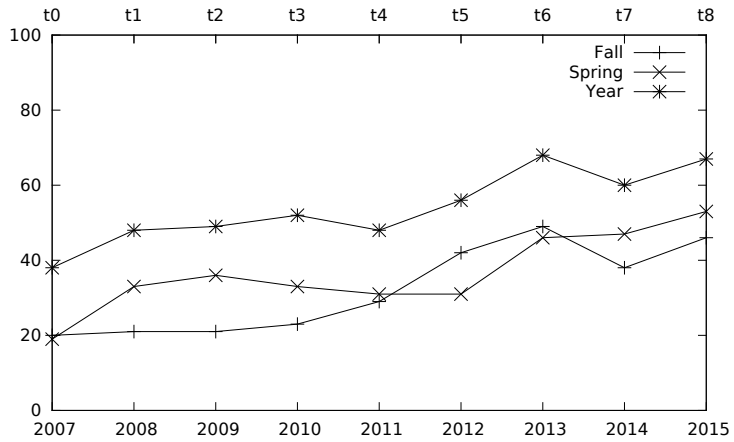

Fig. 2. Percentage of students who pass the course by year (or timestamp): A graphical representation of the 4th, 7th and 10th columns of Table 2. Timestamp t_0 is labeled as 2007 but the data corresponds to the course 2006-2007, and so applies to the rest of timestamps, respectively.

and progressively. Without that background, they got often blocked and frustrated when tackling different or more difficult problems.

- The remedial exam introduced at t_6 increased the pass rate from 41% to almost 50%. This amendment seems to suggest that there is a significant percentage of students close to the passing mark threshold who, with a bit more practice and personalized attention, succeed to pass the CS1 course. The mandatory attendance to lectures may have some influence on that success.
- Introducing new problems in the exams that were not in the course lists at t_7 decreased the pass rate from 50% to 38%. This important decrement reinforces the risks mentioned for the initiatives taken at t_5. The impression of the faculty members is that this decrement may temporal and should improve in the forthcoming semesters.
- With the course diversification done at t_8, the pass rate grew up from 38% to 46%. It seems that tackling new training exercises motivates the students to work harder. This seems to help them to face the exams with a more positive attitude. We observed also a positive effect in the acquisition of programming skills.

Observe, that the pass-rate curves for the fall semester and the academic year follow similar tendencies. The pass rate of the spring semester is in general higher that the one for the fall semester. Only in t_5 and t_6 the rate drops below. However, as it can be seen in Fig. 2 the pass rate in the spring semester shows a different tendency.

Table 3. Workload (in hours) of the tasks of the course per semester.

	Fall								Spring							
	E_t	T_t	L_t	G_t	C_t	R_t	S_t	V_t	E_t	T_t	L_t	G_t	C_t	R_t	S_t	V_t
t_0	24	707	1696	100	33	0	300	75	24	483	1548	91	32	0	300	68
t_1	36	922	2214	196	39	0	300	98	36	555	1777	158	34	0	300	79
t_2	36	931	2236	596	39	0	300	99	36	534	1710	456	34	0	300	76
t_3	30	781	1876	500	35	0	300	72	30	506	1620	432	33	0	300	63
t_4	30	616	2218	591	49	45	300	382	30	277	1332	355	39	45	300	229
t_5	18	615	2214	590	49	10	300	381	18	206	990	264	35	10	300	170
t_6	25	611	2142	573	49	16	300	363	22	184	839	225	35	16	300	143
t_7	37	575	2012	538	48	0	300	340	31	207	947	254	36	0	300	161
t_8	37	590	2066	552	48	358	300	350	31	173	785	210	34	149	300	133

Table 4. Total and per student workload (in hours) of the tasks of the course per semester and year.

	Fall		Spring		Year	
Timestamp	W_t	W_t/N_t	W_t	W_t/N_t	W_t	W_t/N_t
t_0-Kick-off	2937	7.79	2548	7.41	5485	14.55
t_1-Int. of quizzes	3807	7.74	2940	7.44	6748	13.72
t_2-Grad. red verdicts	4240	8.53	3146	8.28	7386	14.86
t_3-Hand-written final ex.	3597	8.63	2984	8.29	6581	15.78
t_4-New degree	4232	8.59	2608	8.81	6840	13.88
t_5-Prob. to hand-in	4177	8.49	1994	9.06	6172	12.54
t_6-Re-evaluation	4080	8.78	1766	9.76	5847	12.57
t_7-Exams with new problems	3851	8.83	1938	9.46	5789	13.28
t_8-Diversification	4303	9.61	1818	10.76	6122	13.67

5 Workload

Our goal in this section is to describe the method used to estimate the workload of the course in each of its timestamped stages, measured as the total number of working hours invested by faculty members. We denote the workload as W_t, where $t \in t_0, \ldots, t_8$. Computing W_t is difficult because every new edition of the course involves slightly different tasks, faculty members with different profiles, different dedication times and different personal efficiencies. Moreover, the faculty members involved in the course also changes from semester to semester. Also the perception about the time invested in each task is different for each faculty member. In order to capture the different type of tasks included in W_t, we have approximated W_t by decomposing it into the following measures:

- E_t: time to design, to test and to prepare exams,
- T_t: time to prepare a theory session,

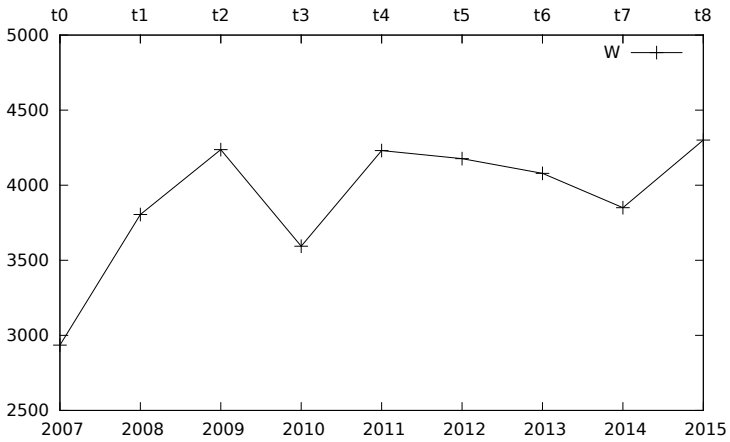

Fig. 3. Workload (in hours) of the course by year (or timestamp) at the fall semesters (i.e., aggregates of Table 3).

- L_t: time to supervise practical sessions,
- G_t: time to mark exams,
- V_t: time to supervise exams,
- C_t: time to coordinate the course,
- R_t: time to redesign the course, and
- S_t: time to maintain the software.

Therefore, for each timestamp t, W_t is conformed by the number of working hours required by the sum of all these tasks, i.e.,

$$W_t = E_t + T_t + L_t + G_t + V_t + C_t + R_t + S_t.$$

To a greater or smaller extend, these quantities are dependent on the number of students taking the course. Observe that we do not include here the working hours required for the initial design of the course and of the design and implementation of the online judge.

In order to estimate W_t, in the fall semester of the course 2014-2015 we conducted a survey among the faculty members who were involved in teaching CS1 since 2006. They were asked to provide us with an estimation of the hours they invest in each of the tasks. The values used for the estimations are the averages over the answers received to that survey. Since we did not have similar information from previous editions of the course, and since many of the teachers who participated in the survey taught CS1 in several editions of the course, 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 as Appx. A. The obtained values are shown in Table 3.

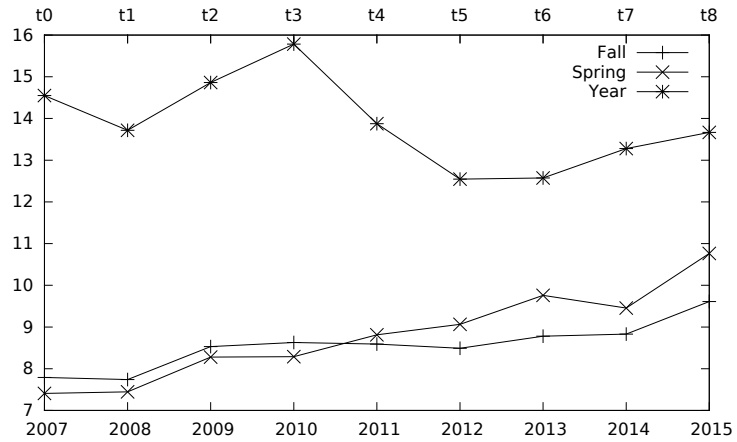

Fig. 4. Hours of faculty workload per student $W_t^s = W_t/N_t$ by year (or timestamp).

Figure 3 shows the evolution over time of the workload of each of the tasks in the fall semester. One can see that the most significant contributions to W_t at each timestamp are practical lectures and theory lectures. In spite of that, the faculty members have the impression that the same does not apply to students. Some students, as the course advances, show a tendency towards not attending lectures.

One can also see that most of the tasks (except G_t and V_t, i.e., marking and supervising exams) are almost constant over time. Both measures increase in time and are the principal reason why W_t also increases. Indeed, this behavior is as expected because the measures introduced along the years are mostly evaluative ones, i.e., directly or indirectly in the form of exams. Therefore, it is natural that they mostly impact the tasks involved in designing, supervising and assessing exams.

Once we have an estimation of W_t, we can calculate the faculty workload per student at timestamp t as

$$W_t^s = W_t/N_t.$$

This measure approximates how many of the working hours of the faculty members are dedicated to each student. In other words, it estimates what is the *cost* of every student in terms of faculty working hours. Table 4 contains the data for both semesters and the sum over the year, and Fig. 4 shows the evolution of this cost over time for the fall semester. As the plot shows, the cost per student increased by an hour from t_0 to t_7, and nearly by one more hour from t_7 to t_8. As we said before, these extra hours mostly corresponds to the increase in grading and supervising exams. Figure 4 shows that an important increment in the cost per student appears at t_2, together with the measure of grading also the programs that obtained a red verdict by the online judge. In spite of that

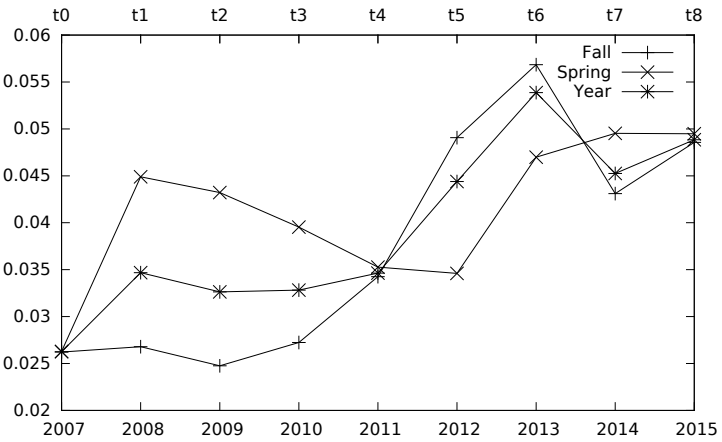

Fig. 5. Productivity Π_t by year (or timestamp).

effort, the t_2 amendment did not have much influence on the pass rate, as we saw before.

6 Method

In this section we go deeper into the analysis of the pass rate and the workload by relating them by means of economics concepts. We present the cost-benefit and the marginal analysis method that we will use for that aim.

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 total workload W_t incremented just by a modest 13% while duplicating the pass rate in the side of students. However, the measures taken did not affect the whole workload but only G_t and V_t, which became tripled. Therefore, the duplication of the pass rate does not seem to justify the triple increment 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 [13]. In our case, being very coarse and with all the safeguards and warnings required, one can see that 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* at timestamp t Π_t as the ratio of these two quantities over time, i.e.,

$$\Pi_t = P_t / W_t.$$

Recall that the S-curve shape in Economics [13] has generally the interpretation that once the inflection point is reached there is no sense to continue increasing the input (i.e., the workload of the course, in our case) because this

increment has no impact on the output. In fact, it is negative. Figure 5 shows the behavior of Π_t from t_0 to t_8 for the fall and spring semester, as well as for the total academic year. Abstracting from short term oscillations, the tendency of the year productivity seems to approach the S-shape. This is also true in the fall semester. However, the irregular behaviour of the spring semester is reflected in its productivity curve, making it difficult to analyze its tendency. Recall that the students conforming the spring semester are those who failed in the previous semester(s) and that strongly conditions and bias the outcome. The results of the spring semesters can not be stand-alone analyzed, but inside the academic year. Specifically, the year productivity curve Π_t for the fall semester and the academic year has four sections that are noteworthy:

1. The first one is from t_1 to t_2 in which it decreases due —as we already mentioned— to the grading of exam problems labeled with a red verdict by the online judge.
2. The second one is from t_2 to t_6 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 third period corresponds to t_6 to t_7 in which Π_t drastically decreases. In this period the workload increased but the number of students that passed the course decreased. This is because the mid-term practical exams are currently composed by new problems that are not known in advance by students. measure
4. From t_7 to t_8 the curve moderately increases again. Even if the workload has increased this seems to be compensated by the increase of the pass rate.

We consider now more closely the impact of the different measures over time. To capture the variation of work among periods we define, for $t \in t_1, \ldots, t_8$,

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

and for the variation of students that succeed on 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 $\Delta_t = \Delta W_t/\Delta P_t$. Making again an abusing use of economics terminology, we call this rate the *marginal gain* at time t of the undertaken measure [13]. We can take into consideration 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 negative undertaken measure. 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 negative option. It may be desirable to decrease ΔW_t but not at the cost of decreasing also ΔP_t. However, if $|\Delta W_t| \gg |\Delta P_t|$ the undertaken measure 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.

Table 5. Variations on the pass rate, workload and marginal gain, per semester and year.

	Fall			Spring			Year		
	ΔP_t	ΔW_t	Δ_t	ΔP_t	ΔW_t	Δ_t	ΔP_t	ΔW_t	Δ_t
$t_0 - t_1$	0.31	-0.05	-0.17	13.94	0.04	0.00	9.36	-0.84	-0.09
$t_1 - t_2$	0.40	0.79	2.01	2.37	0.84	0.35	0.93	1.15	1.23
$t_2 - t_3$	2.37	0.10	0.04	-3.01	0.01	0.00	3.31	0.92	0.28
$t_3 - t_4$	5.91	-0.04	-0.01	-1.70	0.52	-0.31	-3.73	-1.91	0.51
$t_4 - t_5$	12.25	-0.09	-0.01	0.28	0.25	0.89	7.62	-1.33	-0.17
$t_5 - t_6$	8.23	0.28	0.03	14.49	0.69	0.05	12.05	0.03	0.00
$t_6 - t_7$	-11.82	0.06	0.00	0.97	-0.30	-0.31	-7.65	0.70	-0.09
$t_7 - t_8$	8.58	0.77	0.09	6.43	1.31	0.20	6,65	0.39	0.06

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 positive 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).

7 Analysis

In this section we interpret the amendments taken in the CS1 course in terms of the cases described in the previous Section, using the cost-benefit and marginal analysis method. Table 5 shows the values of ΔP_t, ΔW_t and Δ_t over time. We comment on the results for the academic year and add some comments when there is a variation in some of the semesters.

t_1 **Introduction of quizzes** (2007-2008): This measure falls under Case e). Since $|\Delta W_t|$ is very small this was a moderately productive measure. Observe also that for the spring semester this measure falls under Case d) however again under Case e) for the whole year.
t_2 **Grading red verdicts** (2008-2009): This amendment falls under Case c). As we already mentioned this was a negative 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. As before, the positiveness on the

second semester, that fells under Case d), cannot change the tendency of the first semester.

t_3 **Hand-written final exam** (2009-2010): The amendment falls under Case d). So it seems to be a positive measure. Indeed, given that the red verdicts have to be assessed, it is better to have written exams since the time to design and supervise them is lesser. However, this is only true in the context of evaluating red verdicts, not in general. If compared against t_1 then it seems to be a negative measure.

t_4 **New degree** (2010-2011): This is a positive measure that falls under Case e) for the first semester. The measure can be classified under a) for the spring semester and under b) for the whole year. This is the only measure in which the spring semester changes the final tendency. Introducing mid-term exams with a significant weight over the final grade seems to have a positive impact on the pass rate. That compensated the decrease of hours in theory lectures.

t_5 **Lists of problems to hand-in** (2011-2012): This seems to be an outstanding amendment. It falls under Case e). However one has to be prudent with such kind of amendments. There is no doubt that it increased the pass rate 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 C. P. Snow 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."* (See Foreword of [5].)

t_6 **Re-evaluation course** (2012-2013): This can be considered a positive amendment despite being very expensive. It falls under Case d).

t_7 **Mid-term exams with new exercises** (2013-2014): This amendment falls under Case a) for the fall semester and the year and under Case e) for the spring semester. 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. The values for the spring semester seems to confirm the impression that given more time students can adapt their learning to assume this kind of measures.

t_8 **Course diversification**: This is a positive measure that falls under Case c). Students become more interested in the new material and they seem to work harder, increasing consequently the pass rate.

8 Conclusions

In this paper we have described the evolution of the continuous assessment of the CS1 course. This evolution has been defined by the series of measures that have been taken to increase the pass rate since its inception. The successive introduction of these reported measures, as a way to incentive students work,

increased the weight of continuous assessment. However, the pass rate of this massive course has not increased as much as expected. In fact, we have seen that the increase of the faculty workload paralleled the increase of continuous assessment.

We performed a quantitative analysis of the temporal evolution of the pass rate/workload ratio of the evaluative activities as a method to assess the impact of the introduced measures. Our analysis is based on two functions: productivity and marginal gain. We use them to perform a cost-benefit analysis. The proposed method allow us to assess whether a measure should be maintained, tuned or withdrawn, under the general hypothesis 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 it becomes clear that, for some of the adopted measures, the amount of invested resources (faculty workload) did not justify their impact in the pass rate. For instance, the substantial overhead of the measure of grading red verdicts had almost no impact on the passing rate. However, other measures did have a positive impact without increasing the workload, as for example, the weights given to the different exams. Moreover, as all the introduced measures involve continuous assessment, our study shows that the corresponding workload is close to its limit.

The analysis tools proposed in this study provide a way to analyze the effectiveness of new measures. Our findings are valuable for the design of future strategies for this and similar courses. Several pedagogical strategies, around the use of the online Judge as an automated aid to motivate, help and evaluate students, have been introduced in the CS1 course. Some of them have been successfully used also in other courses at our university (CS2, Data structures and algorithms, Algorithms, Functional programming, among others). It would be of interest to perform this kind of analysis for the evolution of the continuous assessment of those courses.

We have focused uniquely on the pass rate/workload ratio, however the scope of our study could be extended in other ways. As an example, taking into account students marks and motivation will provide a finer analysis which might bring more insights in the effectiveness of every measure.

References

1. Arrow, K.J.: The economic implications of learning by doing. The Review of Economic Studies 29(3), 155–173 (1962)
2. Bowles, S.: Towards an educational production function. In: Education, Income, and Human Capital. pp. 9–70. National Bureau of Economic Research (1970)
3. Giménez, O., Petit, J., Roura, S.: Programació 1: A pure problem-oriented approach for a CS1 course. In: Hermann, C., Lauer, T., Ottmann, T., Welte, M. (eds.) Proc. of the Informatics Education Europe IV (IEE-2009). pp. 185–192 (2009)
4. Hanushek, E.A.: Education production functions. In: The New Palgrave Dictionary of Economics. Palgrave Macmillan (2008)
5. Hardy, G.: A Mathematicians Apology. Cambridge (1940), reprinted with Foreword by C.P. Snow 1967. Cambridge University Press, Canto Edition, 1992

6. Ihantola, P., Ahoniemi, T., Karavirta, V., Seppälä, O.: Review of recent systems for automatic assessment of programming assignments. In: Proceedings of the 10th Koli Calling International Conference on Computing Education Research. pp. 86–93. ACM (2010)
7. Martín-Carrasco, F.J., Granados, A., Santillan, D., Mediero, L.: Continuous assessment in civil engineering education — yes, but with some conditions. In: Proceedings of the 6th International Conference on Computer Supported Education, Volume 2, Barcelona. pp. 103–109. SciTePress (2014)
8. Petit, J., Giménez, O., Roura, S.: Jutge.org: an educational programming judge. In: Proceedings of the 43rd ACM technical symposium on Computer science education, SIGCSE 2012. pp. 445–450 (2012)
9. Revilla, M., Manzoor, S., Liu, R.: Competitive learning in informatics: The UVa online judge experience. Olympiads in Informatics 2, 131–148 (2008)
10. Solow, R.M.: Learning from 'Learning by Doing' Lessons for Economic Growth. Stanford University Press (1997), series: Kenneth J. Arrow Lectures
11. Stiglitz, J.E., Greenwald, B.C.: Creating a Learning Society: A New Approach to Growth, Development, and Social Progress. Columbia University Press (2014), series: Kenneth J. Arrow Lectures
12. Tonin, N., Zanin, F., Bez, J.: Enhancing traditional algorithms classes using URI online judge. In: 2012 International Conference on e-Learning and e-Technologies in Education. pp. 110–113 (2012)
13. Varian, H.R.: Intermediate Microeconomics: A Modern Approach (7th Edition). W. W. Norton and Company (2005)
14. Verdú, E., Regueras, L.M., Verdú, M.J., Leal, J.P., de Castro, J.P., Queirós, R.: A distributed system for learning programming on-line. Computers & Education 58(1), 1 – 10 (2012)

A Technical Details

In this appendix we calculate the amount of working hours per task at each stage of our course based on the 14 answers that we obtained by surveying 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 practical, 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:

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

where N_t is the total number of enrolled students at time t. Proceeding similarly for practical sessions and considering that the size of the laboratory groups is of 20 students, and that the preparation of each hour of practical 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 which 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 laboratory 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 verdict 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 two parts. A fixed cost, estimated as 1h per week giving 15 hours, that is $k_0 = 15$. Another part depending on the number of students of each course. We estimate this last amount in one half hour per group of 10 students, then $C_{t_0} = k_0 + 0.5 \times \frac{N_{t_0}}{10}$. 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. As this period corresponds to a start-up, there is no redesign and therefore, $R_0 = 0$.

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 $G_{t_1} = 4 \times 0.3 \times \frac{N_{t_1}}{3}$. The workload of all the other tasks remained the same. There is no redesign, $R_1 = 0$. We also take $k_1 = k_0$

t_2 **Grading red verdicts** (2008–2009): When all the submissions (and not only the green labelled ones) have to be graded G_t was triplicated. $G_{t_2} = 4 \times 0.3 \times N_{t_2}$. We take $k_2 = k_0$ and $R_t = 0$.

t_3 **Hand-written final exam** (2009–2010): At this point the final exam was changed to be a written exam of 3 problems. The exam was organized in only one turn applied to all the students (same exam for all students).

The time to prepare a problem for a written exam is estimated in 2 hours (1 hour less than the time of a practical exam). Therefore $E_{t_3} = E_{t_1} - 2 \times 2 \times 3 + 3 \times 2$. The written exam lasts for 3 hours. Since rooms with place for 40 students where used for a written exam, V_t decreased to $V_{t_3} = 2 \times \frac{N_{t_3}}{20} + 3 \times \frac{N_{t_3}}{40}$. As before $k_3 = k_0$ and $R_3 = 0$.

t_4 **New degree** (2010–2011): The fix cost due to bureaucratic duty in increases from k_0 in an extra half an hour per week plus 2 hours of exam coordination

giving $k_4 = k_0 + \frac{1}{2} \times 15 + 2 = 24.5$. 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 course redesign R_4 increases. As the 15 sessions shrink in 1 hour, a main redesign of the contents is needed. The estimated cost is 3h per session giving $R_4 = 3 \times 15 = 45$. The evaluation system also changed. 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 practical exams lasts 1.5 hours each while the last one for 2.5 hours. The final exam still lasts 3 hours, thus $V_{t_4} = (3 \times 2 \times 1.5 + 2 \times 2.5) \times \frac{N_{t_4}}{20} + 3 \times \frac{N_{t4}}{40}$.

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 laboratory problem decreased to 1.5 hours. Hence $E_{t_5} = 4 \times 2 \times 1.5 + 3 \times 2 = 18$. $C_{t_5} = k_4 + 0.5 \frac{N_{t_5}}{10}$. Finally, the preparation of the lists set up R_5 to 10 hours, so that $R_5 = 10$.

t_6 **Re-evaluation course** (2012-2013): The k_5 increases by 2 coordination hours needed to manage of the re-evaluation, $k_6 = k_5 + 2$. The R_6 is increased by 6h because 3 new large lists plus the corresponding sessions has to be generated, $R_t = 16$. In this case $E_6 = E_5 + 2 \times 2 + 3 = 7$. Moreover we need to add 20 hours to T_t, 60 hours to L_t, 15 hours to G_t and 3 hours to V_t.

t_7 **Mid-term exams with new exercises** (2013-2014): This measure involved the creation of new problems for the mid-term practical exams, instead of taking them from the lists. This increased the time for preparing each problem from 1.5 to 3 hours. Thus $E_{t_5} = 4 \times 2 \times 3 + 3 \times 2 = 30$.

t_8 **Course diversification** In this case $k_8 = k_7$. C_8 is computed as in the previous step. Other cases are similar. The big difference is in the course redesign. First of all, there is a fix part cost due to list redesign. New 15 list has to be build, with a cost of half an hour per list. The lab training of the new list, depends on the number of groups $N_8/10$, on the number of list and is also 1/2 hour. Therefore

$$R_8 = 1.5 \frac{\text{hours}}{\text{list}} \times 15 \text{ list} + \frac{1}{2} \frac{\text{hours}}{\text{list} \times \text{grup}} \times 15 \text{ list} \times \frac{N_8}{10} \text{ grup} = 358.5 \text{ hours}$$

The quantification of this model was provided in Tables 3 and 4.