A context for programming learning based on research communities

Scheila Wesley Martins
University of Coimbra - UC
Department of Informatics Engineering - DEI
Coimbra, Portugal
scheila@dei.uc.pt

António José Nunes Mendes
toze@dei.uc.pt

António Dias Figueiredo
adf@dei.uc.pt

Abstract— **This paper describes a research work that seeks to develop a pedagogical strategy to assist in programming learning, inspired by Mathew Lipman's strategy – conceived to help students maximize their learning through the conscious assessment of their self-efficacy level while they develop a programming study behaviour. We present the results of the preliminary tests on the formalization of the research methodology and the changes implied on the strategy to be tested.**

Programming learning; Research communities; Literary skills; Self-efficacy

I. INTRODUCTION

There seems to be a consensus among teachers and researchers that regards programming learning as a non-trivial activity, as it introduces a new series of cognitive requisites in students' routines, besides the technical requisites. This creates in students the need to change and adapt their way of thinking and manage time in college life to a different reality than the one lived through primary school and high-school. A change in a very short time frame. Among these requisites we may highlight:

1. Problems solving is above all a competence that involves cognitive processes - such as creativity and rationality - from a set of mental meta-abilities (abstraction, inference, deduction, etc), meta-abilities that sometimes come unnoticed and are supported on the exercise developed through their basic literacy skills (like reading and interpreting the description of a problem). Besides, in order that the student may start the aforementioned cognitive processes so that he can use his mental skills to build a solution, he needs to learn how to contextualize the acquired knowledge.
2. The whole understanding of a programming paradigm's requisites is not a trivial activity and implies an intrinsic natural difficulty level [1, 2]. However, the major difficulty in programming learning lies on the ability to abstract the acquired knowledge to solve problems. Abstraction and problem resolution abilities can only be obtained through individual effort and continuous practice. Having well-developed literacy abilities will be of extreme importance to ground the cognitive processes and to support the learning being created.
3. Although computer science is part of the students' daily life from an early age on, the transparency level intrinsic to the relationship user-computer leads them into a false sense of intelligent autonomy of informatics systems. Sometimes it is difficult to make them understand the difference between the notion of systems transparency and their responsibility in the relationship programmer-computer.

The abilities of abstraction and problem solving are underdeveloped among university newcomers from high-school. The lack of excellence in their development is further stressed by the superficiality of their literary skills [3, 4], crucial to a productive academic life and to develop the technical abilities proposed in the syllabi of university courses.

The major problem concerning programming teaching does not lie only on the difficulty students have to abstract and solve problems. There is still the difficulty in finding a way that motivates students to get involved in the course in spite of their handicaps and to not give up trying to overcome the natural barriers inherent to this learning [5]. It is vital to make them realise that the obstacles are surmountable and that the demanded skills may be developed and improved, encouraging them to develop and strengthen a set of essential academic competences so that they may better improve in their studies and in their future professional life.

II. TEACHING TO THINK

Access to education is an evident concern of today's society and of the global economy, with a strong commitment from governments of rich and developing countries in initiatives to improve and offer better conditions for learning in the different stages of the educational system: the enlargement of vacancies in the educational system, the evaluation of the quality of teaching, developing policies of social and digital inclusion, investment in the renovation of classrooms and laboratories with computing equipment and programmes to improve the qualification of teaching staff.

Nevertheless, those reforms have not yet been able to produce a system that can develop towards the "teaching to think" concept. There is nowadays a great number of students, from elementary to high school, who do not develop well enough the different abilities and competences necessary to evolve in a more productive way throughout university years, as shown in the results of the International Programmes of Teaching Evaluation [6].

The growing number of students in the university system has burdened the traditional academic model. In a short timeframe there was a growing demand that made the academia need to change their way of work and educational model, without a previous preparation to adequately respond to that demand.

Facing the urgency in finding ways to manage this situation, the academia often chooses processes that privilege administrative issues (classes format, resources allocation), which is not always the best solution under a didactic point of view. Although there is a need for a renovation in the academic life to better suit the new reality, it is a process that is under development and where the administrative reforms end up influencing the evolution of didactic processes.

A. Critical and Creative Thinknig

Matthew Lipman is an educational theorist who has called the attention from the North-American Academic Community in the seventies with his proposal of Teaching Philosophy to children, we called "Pedagogy of Judgment" (Philosophy for Children – P4C)[7]. It suggests to "teach to think" using the philosophical speeches, proposing teachers to readopt the Socratic teaching as a didactic approach.

His goal is to start in infancy a long term process of development of critical and creative thinking, joining literacy and language acquisition which will last throughout all the formal educational process that the child will live. In that proposal, at the same time the child begins to develop literacy skills, he/she also starts the continuous development of his/her abilities to think, going from the stimulus to the capacity of judging fairly.

To Lipman, issuing a judgment is a behavior that sets the basis for the qualitative development of the most relevant cognitive abilities for educational goals (research, reasoning, information organization and translation), thus making an elementary unit of thought. Judging is the basic cognitive unit for the development of critical and creative thinking, and it is influenced both by criteria (its rational element) and by individual values – the emotional element of its composition.

His proposal defines practical actions which converge with the thinking of John Dewey, Lev Vygotsky and Jerome Bruner, aiming at changing the wrong concept that childhood innocence impedes the child from learning to use reason as a learning tool. His work shows that the lack of motivation to learn must be understood more as a result of the evolution of non-reflective practice of the traditional educational model, rather than the "innate" lack of curiosity from the student [7].

This pedagogical strategy is implemented from a Lipman's redefinition of the concept of learning communities which he called "community of inquiry". Lipman's proposal is an appeal to teachers so that they focus their practices in guiding students in a search for knowledge: motivating them to learn how to consciously identify what they already know and what they need to know not only how to solve a problem but also to be able to think on and assess the quality of a solution.

B. Teaching to think in programming

There is an intense effort from researchers and teachers in trying to understand the reasons that make programming learning be seen as an obstacle for a growing number of students [8, 9].

In fact, the major problem is not language or the programming paradigm, but the difficulty to develop competences that make the student able to contextualize his/her knowledge in order to solve problems. It is important to make the student realise that programming is, above all, a conscious exercise of the mental abilities that are developed and offered in an appropriate context to support the development of several cognitive abilities [10,11].

Some researchers aim at a restructure of curricula [12], but that can only be implemented through the change in the way activities are proposed and to evaluate the progress of the student's learning [13, 14]. There are advances that can be obtained for instance with the regular practice of activities related with research and reflexion for problem solving, abstraction, modelling and the evaluation of the quality of algorithmic solutions [15, 16].

It would then be possible to understand programming learning as a situation that encompasses nuances similar to the context of language acquisition by children. Up to a certain extent, learning how to program requires a change in the way of thinking, according to the paradigm's precepts, just as when one learns a new language.

The relationship of mutual influence between language and thought is evident [17], and resembles to the relationship between language and the programming paradigm. That is why it is relevant to get close to Lipman's approach to programming learning.

Although the pure philosophical speech is not *per se* a natural approach to teaching practice in programming, the methodical and reflective thought that comes from the dialogue inspired in Science Philosophy is rather useful.

Literary skills may thus be reinforced together with programming teaching, as they will assist in the learning of the several stages of abstraction and modelling of an algorithmic solution and in the software development process. Furthermore, as one develops mental abilities, that usually influences the qualitative improvement of the several literary skills (reading, interpreting, summarizing, criticizing, etc), and vice-versa [7].

Thus, rather than just being able to define a methodology of teaching-learning for programming, it would be interesting to be able to establish a set of strategies that show students that solving programming problems is an activity that they are fully capable of accomplishing. It is important to value contexts and establish a dynamics in class that may motivate students to teamwork, giving evidence and making them aware that individual difficulties are surmountable so that they get ready to "learn to think" [18, 19]. This should lead to a higher student commitment to their learning, including behavioral changes that may improve their performance throughout the course.

III. MOTIVATION AND LEARNING

Motivation has a great impact on the individual's cognitive development and is a determinant factor for the individual success in a learning process. It is also among the most fascinating features of the human psyche, having one of its main research origins in Maslow's Motivation Hierarchy [20]. Understanding motivation to learn requires a deep analysis of the socio-cognitive components associated with personality (identification with the institution, Degree, career, accessibility to teaching and the analysis of school success) and the quality of relationships in coexistence environments (social identification and learning approaches) [21].

Along evolution in the formal learning process, from primary school, the student faces several tasks, contexts and learning methodologies. From those experiences the student develops his/her study behavior, finding a learning strategy and the method and activities that best fit his personality, his beliefs and values. Yet, it is not possible to underestimate the impact of the experienced learning methodology, as it influences greatly the type, quantity and quality of the abilities and competences that will be developed. This ends up merging within the study behavior of students of a whole generation.

Motivation presents itself as a domain of crucial impact on learning tools and learning [22], and several theories and instruments have already been described to classify measure, create and maintain students' motivation, especially among children and teenagers [23].

With the advance of research in Distance Education (DE), tools inspired in the John Keller's Motivation Model ARCS (Attention, Relevance, Confidence and Satisfaction) [24], have been widely used in course modelling and e-learning environments.

Formal tools to evaluate motivation measures (comfort, self-efficacy, satisfaction) such as the Inventory of Attitudes and Study Behaviours (IABS/IACHE) [25], are important and seek to assess features related with the students' learning strategies. The IASB is an independent, generic behavioral test with which it is possible to evaluate:

- If a didactic strategy may or may not satisfy a set of learning requisites in a given course;
- To point the existence of changes of attitude from the student regarding his/her academic performance
- To establish statistical parameters of a population, identifying the proportions of cognitive, motivational and behavioral dimensions.

The IACHE encompasses cognitive, motivational and behavioral dimensions, distributed in five sub-scales [25]:

- Comprehensive focus, using reflection and a deep content analysis, which implies for the student a greater effort and time in learning;
- Reproduce focus, a tendency to spend only a minimum effort on a superficial learning, based on memorization and contents reproduction;
- Involvement, or motivation, because this related essentially with requirements of intrinsic motivation
- Organization, it analyzes the indications of establishment some organization level on activities of study;
- Competence personal perception.

Other tools focus on more specific features, such as the case of the Course Interest Survey – CIS and the Instructional Materials Motivation Survey-IMMS) [26], two motivation evaluation tools, created in the framework of the ARCS model. CIS and IMMS may show the levels of attention, relevance, trust and satisfaction among students regarding a given course (pedagogical approach, classes rhythm, teaching practice, proposed activities) and the didactic materials used (textbook, handouts, worksheets).

The analysis of the study behavior through tools like IACHE may provide crucial information on the students' behavior, both individually and as a group, allowing to guide and/or to evaluate the selection of activities, methodologies approach and pedagogical practices for students with specific characteristics.

Although tools like IACHE and CIS have been proposed to measure motivation, there is the need to evaluate the level of acceptation and rejection of students to do the activities [27], as well to establish a motivation measure more closely associated with the aptitude of students to learn to program, which would also allow it to be measured with total independence from the other motivational models observed.

That motivation measure is self-efficacy [28], evaluated from scales directed associated with a self-analysis of the ability or inability of a student to perform a specific task. Self-efficacy scales for programming are a formal tool that may be independently and regularly used [29], and which can aid to maintain the student alert regarding the quality of his/her learning, offering another perspective of the assessment of his/her capacity, different from essays and exams' grades.Units

IV. PROPOSED PEDAGOGICAL STRATEGY

This pedagogical strategy aims at defining a theoretical framework that encompasses a set of recommendations regarding contexts and didactic activities, computational tools and motivational strategies that may assist the teacher in the definition of learning contexts for programming courses, as shown in figure 1. The goal is to try to identify the conditions that may make programming learning more stimulating, minimizing drop-out intentions and making the student learn more and better.

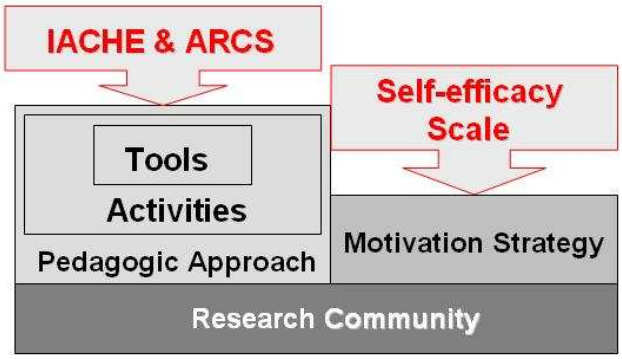

Figure 1. Theoretical scheme of the strategy.

This is a proposal developed under the perspective of learning communities, inspired by a metaphor of Mathew Lipmanp's research communities [7], considered to be a rather relevant abstraction for proposals involving the development of critical thinking [30] and literary skills, and also as a strategy to improve the capacity of solving programming problems among university students [31].

Learning communities are already a renowned perspective, both educational and technological, that allows the creation of contexts in which it is possible to reinforce learning through the association between collaboration activities and knowledge production [32]. Besides, that perspective favours the practice of several literary and mental abilities that will be necessary to students throughout their academic, social and professional lives.

The dynamics of Lipman's Research Communities for programming teaching defines a didactic approach that would allow the teacher to work in several different contexts: Problem Based Learning (PBL) [33], Just-in-Time Teaching (JiTT) [34], hands-on projects [35], games [36] and programming competitions [37] for example.

This is the first foundation of that proposal, i.e., the teacher should decide his/her didactic approach, being able to choose any of the suggested contexts to determine: how and which didactic activities will be developed, and which support tools he/she will use in the course. We suggest these contexts as they have appeared in the literature as well succeeded examples in programming learning.

When choosing the context the teacher should take into account some facts as class size, which can make certain contexts harder than others.

After choosing the context, the dynamics of Research Communities will be developed together with didactic activities planned to strengthen the student's involvement with the process of knowledge acquisition and development of competences to solve problems, through teamwork and the motivation to practice their literary skills in several ways:

- Research activities – which increase the research of content for the solution of problems, e.g.: research questions, evaluation simulations, exercises of discursive evaluation of the algorithmic quality of solutions, practical and/or programming challenges with different complexities, either individual or in group;
- Collaborative/cooperative knowledge production – adoption of methodologies and collaboration tools in reinforcement activities, peer review, motivation for the production of portfolios, webquests, seminars and participation in competitions;
- Distribution of work and tutoring – stimulating the use and recognition of voluntary tutoring between student-student and student-tutor;
- Continuous assessment – regular feedback tasks between teacher, tutors and students on the course evolution.

Programming learning demands the choice of technological tools that can support programming learning and collaborative work. We see as an advantage the use of algorithmic simulation tools and learning support programs adequate to the studied language and paradigm (RoboCod, Alice, BlueJ, o JavaTool e and SICAS, for instance). Competition and Test Platforms like TopCoder, Mooshak and Online-Judge are also welcome.

To stimulate extra-curricular activities and to support the monitoring and continuous assessment tasks, it is advisable that there is the support of a learning management support, such as the Moodle, Blackboard or AulaNet.

The second founding concerns the need to use a supporting strategy for the maintenance of good motivation levels for the students' learning during the community work. Regarding this, it is important that the teacher may be able to identify specific moments to intervene, advise and settle the students' frustrations concerning their performance [38]. The availability and sensitivity of the teacher are crucial to evaluate the students' progress and commitment, as well as to assist in the maintenance of their motivation levels.

Motivational measures such as the comfort level, self-efficacy, confidence, usefulness and satisfaction with the proposed activities should be checked regularly in order to be able to guide the work rhythm and to adequately direct the teacher's efforts for intervention actions, whether in the daily motivation or in the prevention of behaviors that may lead to students dropping out.

A. *Methodology of Pedagogic Strategy Evaluation*

The proposed strategy will use measurements of the usefulness of the proposed tasks along the course as well as some formal psychological tools to evaluate several cognitive aspects related with motivation: a survey on the features of study behaviour through the IACHE tool, an evaluation of the levels of satisfaction with formal aspects of the course development through ARCS, and a measurement of the students' levels of self-efficacy in relation with language and the studied paradigm.

The results of the IACHE test, as they are not *a priori* related to any course or degree, will be used to assess if the use of a certain learning approach has produced any changes in the study behaviour of a given population.

The motivational levels measured by the ARCS model will provide the evaluation of the evolution of the course (contexts, teaching assessment and used resources). For each proposed task there will be a survey to check those measurements at the end of the course.

Finally, a regular application of a self-efficacy scale in programming will be used [39], making the students aware of their competence level to carry out tasks with the studied language and paradigm. That awareness can better guide their studying efforts and stimulate the evolution of the necessary competences for programming learning, identifying the ones which are at an average or rudimentary level.

The evaluation scheme influences the application of CIS test in the middle of the course, and surveys will be made alongside the proposed tasks. Both the IACHE and the self-efficacy scale will be applied in the pre-tests and post-tests scheme, where the IACHE will take place at the beginning and end of the course. The self-efficacy scale will be applied in the middle and end of the course, preferably before the tests of knowledge evaluation.

V. AN EXPERIMENT WITH STUDENTS OF DESIGN AND MULTIMEDIA

Most programming courses of the Department of Infomatics Engineering usually present a high number of enrolled students - between 200 and 300- and a high level of heterogeneity among students in one course, from the several degrees offered in the Faculty of Sciences and Technology of the University of Coimbra (FCTUC), like Informatics Engineering, Industrial Engineering, and Design and Multimedia. The teaching model encompasses:

- Theoretical classes (2 hours): all students enrolled attend a lecture by a professor in a classroom;
- Practical classes (2 hours): students enrolled are divided into classes of up to 30 students to carry out practical lab activities with the same professor of the theoretical class or some other teacher, and:
- Lab practical classes (2 hours): support classes, remedial work or clarification of doubts, non mandatory, given by course tutors.

Applying an experimental strategy in this scenario would be risky and very time consuming. Any change in the pedagogical dynamics and work methodology within this kind of courses would not only imply a more laborious routine for the teacher, but would also raise the chances of failure of the proposed strategy.

Thus we chose to start with a small course, where management and changes control is much easier. The proposed strategy has started being developed in two stages, with students of the Programming course from the Masters Degree in Design and Multimedia (MDM): the first stage carried out from September to December 2008 and the second, still ongoing, started in September 2009. The aim of this course is to offer students a minimum programming knowledge that may allow them to participate in other courses of the degree that require previous programming knowledge.

The course's syllabus was developed according to the dynamics of the Research Communities, and the didactic approach was conducted to provide a practical learning. In the experiments carried out the following activities are used: individual seminars on artistic projects and applications developed with the supporting language, practical group work assignments, discursive evaluation of algorithmic quality from the peer code analysis, producing an artistic portfolio in programming , and continuous assessment.

The exercises and projects proposed involve a need for research, especially the review of algebra and mathematical knowledge, with monitoring from the teacher and motivation for regular qualitative assessment of the assignments accomplished in teamwork.

Bearing in mind the artistic background of the students participating in the experiment, we chose to use the programming language and IDE Processing programming language as a development tool. It was created at MIT by and for graphic designers [40], a tool that expands and facilitates the capacities to develop artistic works with JAVA language.

In the adopted strategy there is no distinction between theoretical, practical or lab practical classes. All classes are spaces for knowledge construction and practical experimentation, making up a total of 6 weekly hours of work. The course holds a class at the DEI's Moodle, where materials are available and some tasks are performed.

The size of MDM's classes made an almost one-to-one student monitoring possible. It has also maximised the opportunity for the teacher to know crucial characteristics of the students, changing the class dynamics to an eminently research approach during the practice of problem solving in groups.

A. *First Stage*

The first stage group included 11 students, mostly recent graduates from the BSc degrees in the areas of design (Multimedia, Industrial or Communication) and architecture. We chose to use the context of visual hands-on projects of growing complexity, as it would facilitate the students' involvement and interest in the creation of varied practical work.

In this stage the original idea was to verify the coherence of the desired approximation to Lipman's approach, for which a continuous assessment system was used, through the fulfillment of several proposed tasks. Moreover, the evaluation included to check: students' satisfaction with their own performance, tasks, materials and classes rhythm through biweekly reflection about the course evolution in the Moodle platform.

Among the positive aspects point out in the records we can highlight the following: research activities and code analysis, team work and class dynamics. Those activities are considered essential for the approach proposed by the Research Communities, enriching it as an interesting approach for programming teaching. Surprisingly, the peer evaluation tasks and the continuous assessment, as well as the artistic contexts proposed in group assignments, are positively evaluated although with a frequency way below expected.

Amid the most scored aspects in the evaluation of negative points, we highlight: the mathematical examples used during the arrays content, the first challenge of the course involving animation programming of a circumference mathematical model and the lack of individual commitment from students. Among the least scored negative points we underline: the delay in the teacher's feedback, classes' schedule, features of the Processing tool and OOP.

By the end of the course students answered an 8 questions interview related with the perspectives of the ARCS model, in which they presented their evaluation on the course, as well as aspects related with their school trajectory.

The most interesting answers were:

1. What do you consider to stimulate your interest in a course: the syllabus (25%), practical tasks (25%), your individual interest on the subject (25%), when you feel curious on the subject (18%). Only 7% consider the usefulness of the course as important;
2. Do you consider as positive the experience in courses where: there was a clear practical feature (25%), you liked the contents (21%), you felt inspired by the teacher (17%), you felt that the contents were interesting/useful (14%);
3. Do you consider as negative the experience in courses where: teachers were not motivated, with inconvenient, derogatory and/or authoritative attitudes (25%), the contents were not interesting to you (25%), there was a theoretical and expository prevalence (17%) you considered them useless (10%). Only 3% reported a failure in their own studying time management regarding studying.;
4. How do you manage your motivation to learn: in uninteresting activities you do not dedicate more than the minimum required (32%), you believe it is natural to feel unmotivated in certain periods (26%), you try to finish an uninteresting task as soon as possible (21%). Only 3% considered the options of: studying in advance to prevent accumulation of study material or to ask for help to study;
5. What grabs your attention: the usefulness of the task (19%), the teacher-student relationship (19%), the practical characteristics of the course (18%), the teacher's availability (16%), didactics (16%), your curiosity (7%);
6. You feel confident in a course when: you know something on the subject (40%), you trust on the teacher's knowledge (28%), you are interested and confident in the usefulness of the course's subject (12%), you trust the evaluation system used by the teacher (8%);
7. For a task to be interesting, it depends on: adopted context (30%), the clear relationship between theory and practice (26%), empathy with the teacher (26%). Nevertheless, 18% believe that some courses are naturally uninteresting, no matter what they propose;
8. The position of the Programming course in case a ranking concerning the interest on the course was organized: first place (25%), second place (33%), third and fourth place (18%) and fifth place (10%);

The result of the interviews corroborates with the aspects identified in the bi-weekly reflections analysis, with some pleasant surprises:

- Of the five courses attended by students in that semester, Programming and Internet Technologies were the ones that caused more negative expectations due to the frustrating experiences many say to have gone through during their BSc;
- The reports of good experiences with courses denoted the importance of the relationship between didactics and the teacher's posture. They have mentioned as positive experiences in courses where they did not feel attracted by the contents, while they felt inspired by the teacher;
- They have considered Programming a pleasant surprise among the courses they had attended, highlighting the way courses were conducted, the tool which was used and the evaluation process;

Only one student didn't manage to pass the course and the teacher was pleased not only with the results obtained but also with the dynamics of the strategy, and although he recognizes that there is an increase in work, he believes that the way to work is more prolific.

This experience was the first approximation accomplished for the definition of the strategy here described, and it is particularly important to clarify under which foundations it should evolve and which adjustments should be made. Both the feedback and the constant monitoring by the teacher have shown to be extremely important, as they have allowed a better evaluation of the quality and acceptability of the implemented activities.

It has also made possible to carry out on time actions that were adequate for the classes rhythm and the proposed tasks, to address issues of commitment and dispersion, and especially to

be able to gain back students with major tendencies to drop out the course.

It also made clear the need to associate a motivation measure, self-efficacy –specifically connected with the language and the programming paradigm used, to support the process of keeping the students committed during the course. This also establishes a change in the evaluation of the experiment results, adopting formal instruments and parameters to evaluate the several components.

B. *Second stage*

There are 12 students in this group, one of them from stage one. Similar to the previous class, most students are recent graduates in the area of arts and design, and only two are student workers.

In this stage we analyse JiTT's potential, as it makes available a set of tasks that are tightly connected with the goals proposed by the Research Communities. On the second stage experimentation we choose to use once again the context of hands-on projects with the creation of a task inspired in JiTT challenges [41]. Although JiTT offers better opportunities for the development of research activities to a programming course [38], we bet on the positive context of hands-on projects received by first stage students.

The context based on hands-on projects and the tasks list accomplished in the first stage have been kept. One of the changes that were incorporated is a change in the organization and alignment of the syllabus' contents and in the evaluation method. There has been an effort so that the introduction of Object Oriented Programming (OOP) concepts is made earlier. Three new tasks have been introduced: individual programming challenges, mini-tests simulation and mini-tests.

Inspired in the JiTT proposal, small programming challenges have been planned in specific points of the course's syllabus, as a way to stimulate individual work, especially outside the classroom. These challenges include a self-evaluation component concerning the merit of the accomplished programming, making the student used to critical assessment and to the exercise of technical competences in software development.

The mini-test simulation is the only non-scored task and it was prepared so that the student may live the mini-test experience in less stressful conditions. The results are corrected and given back to students in a way they may understand their comprehension level of the course. Bad results will be presented in a way to motivate study outside the classroom. The inclusion of mini-tests does not imply a change in the rhythm or in the approach of the adopted didactics, and it is another opportunity to evaluate individually the level reached by students.

The evaluation methods included in the strategy have been are being put into practice in this second stage. In September we applied the IACHE pre-test, in October the Processing self-efficacy scale pre-test, due to academic events, the CIS test and surveys were postponed to November. The results of the pre-tests performed are presented and commented as follows:

a) *IACHE results*

The IACHE survey is divided into three parts, from which only the first one was considered for the analysis. It includes 44 statements. Groups of questions analyse each of the five cognitive dimensions whose answers vary in intensity from 1 (totally disagree) to 6 (totally agree). The score of each cognitive dimension is obtained by the sum of the answers to the questions for that dimension. The reference values and the average point for the comprehensive and organization focus are given by (1) and all other dimensions given by (2).

$$10 < \overline{X} < 60, \; with \; x_m = 35 \tag{1}$$

$$8 < \overline{X} < 48, \; with \; x_m = 28 \tag{2}$$

Figure 2. Summary of the descriptive statistic indicators of the IACHE pre-test

		comprehen-sive	reproduce	personal perception	involvement	organizing
N	Valid	12	12	12	12	12
	Missing	0	0	0	0	0
Mean		42,58	29,92	21,50	36,58	35,00
Median		42,50	29,50	23,00	36,00	33,50
Std. Deviation		4,078	4,999	4,462	4,621	6,495
Variance		16,629	24,992	19,909	21,356	42,182
Minimum		35	24	13	28	23
Maximum		49	39	30	44	47

At this point of the research the average values observed in the analysed cognitive dimensions, figure 2, show a rather high involvement level of students, who demonstrate a great predisposition for the accomplishment of the course's tasks. Surprisingly, the average of the comprehensive level was higher than the reproductive level, which can be a reflex of the organization sense shown by students. The low level of personal perception is a good indicator, but the great discrepancy between the maximum and minimum scores of the sample for that specific dimension warns the teacher on the aspects that involve self-efficacy for students who present a very high average value among this population.

There was another analysis carried out, assembling the groups of answers in the dimensions comprehensive focus, reproductive focus, motivation and organization in three intensity levels: low (answers 1 and 2), average (answers 3 and 4) and high (answers 5 and 6). That relationship was used to analyse all dimensions except the personal perception. For this dimension the analysis is carried out in the opposite way: low (answers 5 and 6) and high (answers 1 and 2).

The analysis by answer level in each dimension presented in table 1 endorses the analysis of the average value, i.e, this is an optimistic result in the sense that shows a predisposition of the group for learning, just 3% below the sample of the low level. Success expectations may be slightly larger than the apprehension of the course's contents. Although students do not show a very high sense of organization, there is evidence that many of them seek to develop study strategies directed to comprehension at a deep level. The low percentage shown by the low level of the comprehensive focus is rather significant.

Moreover, the fact that this population shows a high sense of capacities is positive, as students with low self-efficacy expectations tens to invest a lower effort in studying activities. Only 9% of the population is at the higher level of personal perception.

The most significant efforts of the proposal should be orientated towards the recovery of students who present more answers at the low level, and the improvement of those at the intermediate level.

TABLE I. IACHE DIMENSIONS INTENSITY LEVELS ANALISE

IACHE Dimension	Intensity Levels (%)		
	low	*average*	*high*
comprehensive	4	52	44
reproductive	20	47	34
personal perception	56	35	9
involvement	3	42	55
organization	23	59	18

The comparative analysis of scores and more specifically of the dimensions intensity level will only be possible after the conclusion of the post-test, at the end of the course.

b) Self-efficacy Scale in Processing

The scale used, translated and adapted from a scale for JAVA [42], includes 32 statements related with tasks concerning the tool, paradigm and problems solution, answered according to the intensity of the level of confidence: 1 is for totally unconfident and 7 for totally confident. The scale score is given by the sum of the answers.

$$32 < \overline{X} < 224, with\ x_m = 128 \quad (3)$$

The analysis of self-efficacy follows the same process adopted by the IACHE test, where the reference value and the average point for this population is presented in (3). The data of the descriptive statistics are presented, according to figure 3.

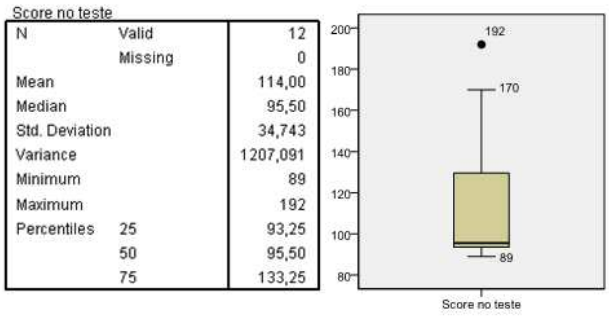

Figure 3. Results of the self-efficacy scale in processsing

The descriptive analysis of the average and the BoxPlot chart shows that there is a large number of individuals whose sense of self-efficacy is very low, since 50% of the population is concentrated on a small amplitude interval between the x_{min} (89) and the median (95,5). The third quarter is a value close to the expected average (128) and thus approximately 75% of the population presents a lower than or equal score to that reference value.

In spite of that statistics result, a more clear conclusion for the studying orientation goals is obtained from the analysis of the intensity level. This analysis reveals a less negative situation, since 63% of the population is concentrated on the average level of self-efficacy (answers 3, 4 and 5), while only 25% are at the low level (answers 1 and 2). Hence, we assume that only one quarter of the sample students consider themselves incapable of accomplishing minimum tasks with the tool, denoting a level of insecurity that makes is suspect that a bad future performance in the mini-tests may actually discourage the student up to the point of no longer doing any effort in the course.

Since self-efficacy is a type of confidence measure that only gives significant modification results when it is at the low level, the teacher's attention should be reinforced towards recovery of students with answers in the low level. Students in the average level may receive the results as an orientation on where to invest their studying efforts, encouraging a more positive expectation of their performance in mini-tests.

It is important to notice and check if the 12% of students who are at the upper level are not too optimistic with the level of their knowledge. In any case, one should not mistake the self-efficacy result with the enthusiasm to learn, checking if, in spite of the positive result, the student is not showing a low commitment with the tasks of the course. All in all, the teacher should adopt intervention measures for guidance and motivation of studying efforts, frustration management and commitment.

As happens with IACHE test, the comparative analysis of scores and more specifically of the dimensions intensity level will only be possible after the conclusion of the post-test, at the end of the course.

CONCLUSION

The general evaluation of the first experiment was considered positive either by students as well as by the course teacher, although there wasn't the full support of all formal cognitive evaluation instruments currently proposed by the strategy. Its accomplishment was particularly important to guide the necessary changes and to plan the second test stage experimentation. It has shown the need to use formal evaluation tools and not only interviews, to validate the strategy's results.

Part of the instruments and premises proposed by the described strategy are also being used to carry out voluntary work, offering programming support to all students of the Faculty of Sciences and Technology of the University of Coimbra (FCTUC), through the Computer Programming Supporting Class. This initiative was backed up by the Faculty's Supporting Office for Portuguese-Speaking Foreign Students, originally aimed at assisting students from Portuguese-speaking African Countries (PALOPs) to

strengthen and develop their minimum competences in problem solving and programming. Having first taken place between February and June 2009, the goal is that these students may have better results in the programming courses of FCTUC. This initiative has currently broadened the access so that all FCTUC students interested in this kind of support may be included.

The changes proposed by the strategy also seek to sensitize faculty members and administrative staff of DEI and FCTUC, showing that there are changes which can be implemented in the organization of programming courses, with positive expectations for the quality of learning and more efficacy in the allocation of physical and human resources. In order to pass from reflection to action, it is important that the results of the proposed changes can be formally documented in a set of measures that have been previously tested in contexts as close as possible to the current reality.

ACKNOWLEDGMENT

The authors would like to thank the students enrolled in the Programming course of the Masters Degree in Design and Multimedia for their participation in the experiments.

REFERENCES

[1] B. C. Wilson and S. Shrock, "Contributing to success in an introductory computer science course: a study of twelve factors," SIGCSE Bull., vol. 33, no. 1, pp. 184-188, 2001.

[2] S. Wiedenbeck, "Factors affecting the success of non-majors in learning to program," in ICER '05: Proceedings of the First international workshop on Computing education research, (New York, NY, USA), pp. 13-24, ACM, 2005.

[3] D. Demerer, "Improving the learning environment in CS1 - experiences with communication strategies," SIGCSE Bulletin, vol. 25, pp. 31-38, September 1993.

[4] C. Bennett and T. Urness, "Using daily student presentations to address attitudes and communication skills in CS1," in SIGCSE '09: Proceedings of the 40th ACM technical symposium on Computer science education, (New York, NY, USA), pp. 76-80, ACM, 2009.

[5] E. Roberts, "Strategies for encouraging individual achievement in introductory computer science courses," SIGCSE Bull., vol. 32, no. 1, pp. 295-299, 2000.

[6] N. B. Dohn, "Knowledge and skills for pisa -assessing the assessment," Journal of Philosophy of Education, vol. 41, pp. 1-16, februere 2007.

[7] M. Lipman, Thinking in Education. Cambridge University Press, 1st ed., 1991.

[8] T. Jenkins, "On the difficulty of learning to program," in 3rd Annual Conference of Learning and Teaching Support Network of Centre for Information and Computer Science LTSN-ICS, (United Kingdom), pp. 27-29, Loughborough University, The Higher Education Academy, Agosto 2002.

[9] E. Lahtinen, K. Ala-Mutka, and H.-M. Jaarvinen, "A study of difficulties of novice programmers," in ITiCSE '05: Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education, (New York, NY, USA), pp. 14-18, ACM Press, 2005.

[10] J. T. Havill and L. D. Ludwig, "Technically speaking: fostering the communication skills of computer science and mathematics students," in SIGCSE '07: Proceedings of the 38th SIGCSE technical symposium on Computer science education, (New York, NY, USA), pp. 185-189, ACM, 2007.

[11] H. A. Etlinger, "A framework in which to teach (technical) communication to computer science majors," SIGCSE Bull., vol. 38, no. 1, pp. 122-126, 2006.

[12] J. Mead, S. Gray, J. Hamer, R. James, J. Sorva, C. S. Clair, and L. Thomas, "A cognitive approach to identifying measurable milestones for programming skill acquisition," in ITiCSE-WGR '06: Working group reports on Innovation and technology in computer science education, (New York, NY, USA), pp. 182-194, ACM, 2006.

[13] S. Kerka, "Techniques for authentic assessment. practice application brief.," ERIC Clearinghouse on Adult, Career, and Vocational Education, Columbus, OH., 1995.

[14] M. K. Hamza, B. Alhalabi, and D. M. Marcovitz, "Creative pedagogy for computer learning: eight effective tactics," SIGCSE Bull., vol. 32, no. 4, pp. 70-73, 2000.

[15] JP. Denny, A. Luxton-Reilly, and B. Simon, "Evaluating a new exam question: Parsons problems," in ICER '08: Proceeding of the fourth international workshop on Computing education research, (New York, NY, USA), pp. 113-124, ACM, 2008.

[16] J. Bennedsen and M. Caspersen, "Assessing process and product - a practical lab exam for an introductory programming course," pp. 16-21, Oct. 2006.

[17] G. Wells, "Language and education: Reconceptualizing education as dialogue," in Annual Review of Applied Linguistics of Oxford (UK), vol. 19, pp. 135-155, 1999S

[18] M. Lindholm, "Object-orientation by immersion: Teaching outside the cs department," in Report of ECOOP 2004 Eighth Workshop on Pedagogies and Tools for the Teaching and Learning of Object Oriented Concepts, (Berlin - Heidelberg), pp. 153-171, Springer-Verlag, 2004.

[19] G. Stegink, J. Pater, and D. Vroon, "Computer science and general education: Java, graphics, and the web," SIGCSE Bull., vol. 31, no. 1, pp. 146{149, 1999.

[20] A. H. Maslow, Motivation and Personality. NY: Harper, 1st ed., 1954.

[21] M. V. Abreu, Cinco Ensaios sobre a Motivação. Livraria Almedina, 2nd ed., 2002. 57-96.

[22] P. C. Blumenfeld, E. Soloway, R. W. Marx, J. S. Krajcik, M. Guzdial, and A. Palincsar, "Motivating project-based learning: Sustaining the doing, supporting the learning," Educational Psychologist, vol. 26, pp. 369-398, June 1991.

[23] S. de C. Martinelli and D. Bartholomeu, "Escala de motivação acadêmica: uma medida de motivação extrínseca e intrínseca," Avaliação Psicológica, vol. 6, pp. 21-31, junho 2007.

[24] J. M. Keller, "First principles of motivation to learn and e3-learning," Distance Education, vol. 29, pp. 175-185, August 2008.

[25] S. Monteiro, R. M. Vasconcelos, and L. S. Almeida, "Rendimento acadêmico: Influência dos métodos de estudos," in Atas do VII Congresso Português de PsicoPedagogia, Universidade do Minho, 2005. Braga.

[26] Keller, J.M. (2010 In Press), Motivational Design for Learning and Performance: The ARCS Model Approach. New York: Springer.

[27] H. Margolis. "Student Motivation: A Problem Solving Focus,"in Reading Disabilities: Beating the Odds, vol. 84, H. Margolis and G. G. Brannigan (Eds.), Reading2008 & Beyond, , July 2009.

[28] A. Bandura, "Self-efficacy: Toward a unifying theory of behavioral change," Psychological Review, vol. 84, pp. 191-215, March 1977.

[29] V. Ramalingam, and S. Wiedenbeck, "Development and Validation of Scores on a Computer Programming Self-Efficacy Scale and Group Analyses of Novice Programmer Self-Efficacy," Journal of Educational Computing Research, vol. 19, No.4, pp367-81, 1998

[30] W. G. Huitt, "Critical thinking: An overview," in Critical Thinking Conference in Barnesville, GA, (http://chiron.valdasta.edu/whuitt/col/cogsys/critthnk.html), Gordon College, may 1998.

[31] S. W. Martins, Um modelo computacional de apoio ao desenvolvimento do pensamento crítico. PhD thesis, Universidade Federal de Santa Catarina - UFSC, Brasil, Maio 2005.

[32] J. E. Minkler, "ERIC review: Learning communities at the community college," Community College Review, vol. 30, no. 03, pp. 46-63, 2002.

[33] E. Nuutila, S. Törmä, P. Kinnunen, and L. Malmi, "Learning programming with the PBL method experiences on PBL cases and tutoring," pp. 47-67, 2008.

[34] G. M. Novak, A. Gavrin, and C. Wolfgang, Just-in-Time Teaching: Blending Active Learning with Web Technology. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1999.

[35] S. P. Linder, D. Abbott, and M. J. Fromberger, "An instructional scaffolding approach to teaching software design," J. Comput. Small Coll., vol. 21, no. 6, pp. 238-250, 2006.

[36] E. W. G. Clua, "A game oriented approach for teaching computer science," Anais do XXVIII Congresso da SBC, pp. 10-19, July 2008.

[37] F. Manne, "Competing in computing (poster session)",in ITiCSE '00: Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSE conference on Innovation and technology in computer science education, (New York, NY, USA), p. 190, ACM, 2000.

[38] H. Margolis and P. McCabe, "Improving Self-Efficacy and Motivation: What to Do, What to Say," Intervention in School and Clinic., vol. 41, no. 4, pp. 218-227, March 2006.

[39] V. Ramalingam, D. LaBelle, and S. Wiedenbeck, "Self-efficacy and mental models in learning to program," SIGCSE Bull., vol. 36, no. 3, pp. 171-175, 2004.

[40] I. Greenberg, Processing: Creative Coding and Computational Art. Friendsof Ed., 1st ed., 2007.

[41] T. Bailey and J. Forbes, "Just-in-time teaching for CS0," SIGCSE Bull., vol. 37, no. 1, pp. 366-370, 2005.

[42] Askar, P. and DaVenport, "An Investigation of Factors Related to Self-Efficacy for JAVA Programming Among Engineering Studentes," The Turkish Online Journal of Educational Technology – TOJET, vol. 8, Issue 1, Article 3, ISSN: 1303-6521, janeiro, 2009.