6-2012

Also Your Job to Learn! Helping Students to Reflect on their Learning Progress

Stella Kakavouli
Wellesley College, skakavou@wellesley.edu

P. Takis Metaxas
Wellesley College, pmetaxas@wellesley.edu

Version: Post-print

Also *Your* Job to Learn! Helping Students to Reflect on their Learning Progress

Stella Kakavouli
Wellesley College
Computer Science Department
Wellesley, MA02481, USA
skakavou@wellesley.edu

Panagiotis T. Metaxas
Wellesley College
Computer Science Department
Wellesley, MA02481, USA
pmetaxas@wellesley.edu

ABSTRACT

Most of the time, the pedagogical analysis on student learning is focused on the efforts of the teacher. There is no doubt that teacher actions can have a profound positive effect on student learning. In this paper, we discuss what students can do to increase their learning outcome. In particular, we describe an instrument that facilitates students' reflection of their own learning. Although the use of reflection on learning is not novel, its use in CS education has not been explored extensively. Our system was developed as a result of an experiment that we performed in a CS2 course, successfully giving students opportunities to take control of their own learning process. One of the attractive features of our system is that it is relatively easy for the instructor to set up and monitor the students' progress. In our evaluation of the system, we found that it can increase significantly the students' programming confidence at the end of the course.

Even though this paper studies the effectiveness of introducing the "reflection questionnaires" instrument in a CS2-type course, it can be applied to most other courses in a curriculum. We provide the material we used in our experiment so that they can be adopted by educators.

Categories and Subject Descriptors

K.3.2 [**Computer and Information Science Education**]

General Terms

Learning Methods, Reflection, Student learning, CS-2 course

1. INTRODUCTION

> By three methods we may learn wisdom:
> First, by reflection, which is noblest;
> second by imitation, which is easiest; and
> third by experience, which is the bitterest.

The quote above, attributed to *Confucius*, served as our inspirational motto for the work we describe below. Most of the time, Computer Science Educators in a programming course, focus on what *they* can do in order to improve the learning experience of their students. We design our lectures to provide a preview of the information our students need to master in order to be prepared for what lies ahead (that's the *imitation* part). We create laboratory exercises and other hands-on activities where the students can practice what they learned in the lectures (the *experience* part).

In a programming course, one can argue, attending a lecture is the easiest for students, while writing, compiling, running and debugging a program during labs (and homework problems) is the bitterest. But how about the first part that Confucius is referring to, *reflection*?

1.1 Prior Work

Educational researchers in the past have been primarily concerned about how to design well-tuned course and lab materials, (too many to list here without bias, but easily found throughout the SIGCSE publications [1]), and how to help students with different learning styles learn in an introductory computing course (e.g., [2, 8]). However, every student can benefit from monitoring his or her learning progress through reflection. Although the use of reflection on learning is not novel, its use in CS education has not been explored extensively.

While the benefits of learning by reflection are known to psychologists for a long time (e.g., [7]), not much work has been done in specializing its benefits in introductory Computer Science courses. An important contribution is the work of [3], which was further developed into a system that facilitates student reflection [6]. Another approach, which requires students to file an experience report that contains a reflective part is described in [9]. While these approaches are significant, they require a bit of teacher preparation and involvement, and they do not appear to be widely used by the Community.

1.2 Brief Description

In this paper we describe a system of reflective questionnaires we designed and implemented using Google Documents during the recent offerings of a typical CS-2 Data Structures course[1]. Briefly, our system works as follows: Af-

[1]We also designed and used an earlier version of the questionnaire for a Multimedia Programming course. This paper refers to the final CS-2 version that evolved out of our prior experiments.

ter each of the eight programming assignments, we required our students to complete a short "Homework Assessment" questionnaire that helps them reflect on their learning process. Summaries of the answers of some of the questions were shared with the whole class at the time the homework was due, providing extra incentive for them to complete it. We also gave a shorter version of the questionnaire after each of the three midterm exams and a different, evaluative, questionnaire after the final project. At the end of the semester, and before the project questionnaire, we emailed back to our students their own responses to the 11 reflection questionnaires they completed during the year. As largely a result of this process, we saw significant increase in confidence in their programming abilities at the end of the course. One of the important characteristics of our system is that it is very easy to create and maintain and can be deployed, even in large classes.

The remaining of the paper is organized as follows: The next section 2 describes the theory behind the effectiveness of reflection on one's learning process. Section 3 describes our system in some detail. The last section 4 has our conclusions.

2. THE NEED FOR REFLECTION

Like many teachers, we often tell our students that in order to learn, it is not enough for us to teach; they also need to want to learn. But, while we, teachers, do a lot to support our teaching, including carefully preparing lectures, review sessions, lab material, homework assignments, etc., we leave the second part up to them. After all, what can a teacher do about a student that does not want to learn? This, however, is not the right question. We are rather concerned about the student who wants to learn, but has not discovered the importance of monitoring and reflecting on his own learning.

The benefits of teaching in a one-on-one basis are well known. With carefully chosen questions, the teachers facilitates learning through student reflection. The student is invited to provide answers to the questions, effectively discovering knowledge through a reflective examination of her own answers.

Unfortunately, despite its effectiveness, this method does not scale up to be used in a classroom setting. Instead, in such a setting teachers usually employ the lecture teaching style. The teacher can still come up with good questions in the spirit of the Socratic method, but the questions are aimed at no particular student – they are aimed at the class as a whole. This is quite tricky to do consistently and does not guarantee that every student will engage in reflectively examining his own learning progress.

Lecturing, therefore, has its limitations. It is not as effective, but it is a very efficient teaching method as it can be applied in a large auditorium. The cost is, however, that the reflective part of the student learning is weakened, sometimes even lost.

A couple years ago, the report entitled "How People Learn: Brain, Mind, Experience, and School" [4], published in 2005 by the the National Research Council, caught our attention. This report synthesizes the recent research literature on learning and focuses on three fundamental and well established principles of learning. The authors argue convincingly that the following three principles are particularly important for teachers to understand and incorporate in their teaching:

1. Students come to the classroom with preconceptions about how the world works. If their initial understanding is not engaged, they may fail to grasp the new concepts and information, or they may learn them for purposes of a test but revert to their preconceptions outside the classroom.
2. To develop competence in an area of inquiry, students must (a) have a deep foundation of factual knowledge, (b) understand facts and ideas in the context of a conceptual framework, and (c) organize knowledge in ways that facilitate retrieval and application.
3. A "metacognitive" approach to instruction can help students learn to take control of their own learning by defining learning goals and monitoring their progress in achieving them.

It was the third principle that caught our attention, since we felt that it was an area that was not adequately incorporated in our teaching.

3. REFLECTION QUESTIONNAIRES

As a result of our early discussions, we decided to design and employ the system we describe in this section. We wanted to produce a simple system that includes an instrument that would be reasonably easy to adopt. For us, the teachers, it should be easy to deploy and for the students easy to follow. In addition, we wanted something that contains both qualitative and quantitative components so that we can measure its success relatively easily.

3.1 Educational Context

We are teaching a typical CS-2 class at a 4-year Liberal Arts College. Our classes have usually between 18 and 25 students, though the instrument we describe can be adapted easily for much larger classes. In our CS-2 course we are using Java, which is also the language used in our CS-1 course.

Our students come to class mostly, but not exclusively, from our introductory CS-1 course. A few students come to our CS-2 having scored a 5 in the AP course, and a few others from another introductory course aimed at students who want to major in the Sciences. This class is using MATLAB as its programming language and the students are encouraged to learn some Java programming on their own.

In previous years we had seen two trends that we wanted to reverse. First, despite our strong encouragement, our students were not likely to collaborate in homework assignments, and we are strong believers that collaborating in a pair-programming style [10] can positively enhance their experience.

Second, our students were leaving CS-2 with mostly the same level of confidence in their programming abilities as when they entered. They definitely had more experience

with programming, but they did not feel that they had become substantially better programmers than when they entered – even though we, as teachers, believed they were.

In previous years, we used an entry questionnaire which was asking about their familiarity with an array of concepts and Java keywords that we expected they had learned in the past. We thought that knowing where our students stand, would help us adjust the course to better cater to their needs. Even though the entry questionnaire was useful in this respect, we found it to be disheartening to our students. The reason is that it emphasized their non-familiarity with several concepts. After tampering with it, we found that the following two questions were enough to provide us with the information we needed, without being discouraging to the students.

- How comfortable are you that you can write a short stand-alone program from scratch?
- How comfortable are you that you can write the code for a small class (with basic constructors, instance vars, getter/setter methods)?

In their responses, they could select one of the following four answers: I am an expert; I am very comfortable; I am somewhat comfortable; I am not at all comfortable. Their responses from the last semester are shown in Figure 1. One would hope that students coming in a CS-2 course would feel at least very comfortable in writing a short program and a basic class. However, half of our students would not feel very comfortable. It is clear that these answers reveal low programming confidence for the majority of our students.

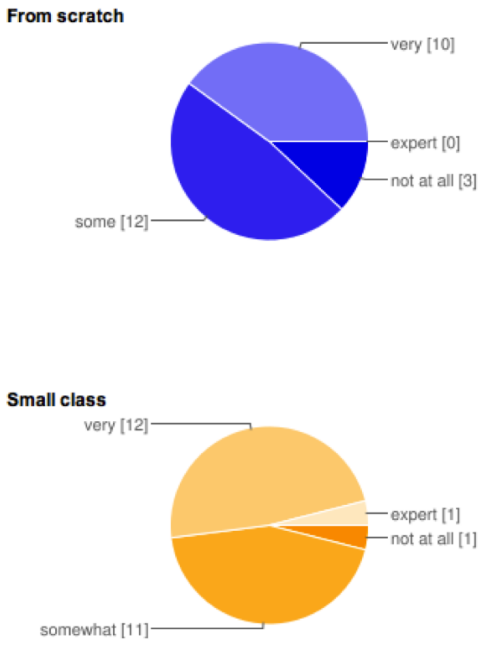

Figure 1: Responses of an incoming CS-2 class on perceived level of comfort in writing a stand-alone program from scratch (top) and a small class (bottom).

Perceptions of competence are important, however. So, we were very happy at the end of the semester, when our students felt that they had made great progress on their confidence level as programmers. As one can see in Figure 2, less than 10% of the students felt that their confidence in programming had remained the same (or decreased), while more than half felt that it had greatly increased. While the course remained greatly unchanged from previous offerings, (same lecture and lab material covered, same set of slides, same textbook, same requirements and same instructor team), we also found the average final grade to increase in comparison to previous semesters. So, both their performance and their perception of programming abilities had improved. [2]

We partially attribute these changes to the introduced instrument that helped the students reflect on their learning. We describe the details of the instrument next.

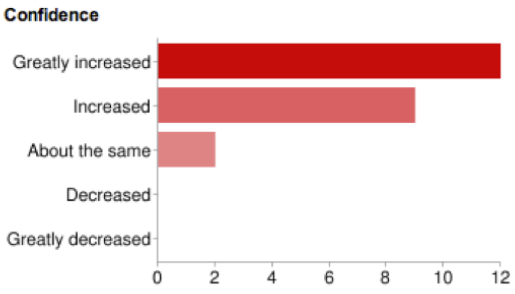

Figure 2: Perceived changes to the level of confidence as programmer, from the beginning to the end of the semester. The x-axis indicates the number of students.

3.2 Details of Implementation

As mentioned, we introduced a sequence of questionnaires, aiming to help students monitor their learning progress during the semester. As an incentive, completing the questionnaires would give students a small credit, which was incorporated in the usual "class participation" credit we always

[2]Since we had a single class section, we could not create a control group for comparison. However, we compare this semester's results to offerings from previous semesters of the course that remained largely unchanged.

had. The questionnaires were given along with each of the 8 homework sets, while a smaller version of the questionnaire was given along with each of the three midterm exams.

A sample questionnaire associated with a homework, requires less than five minutes to complete, and has the following questions.

---

1. How challenging did you find this homework?
2. *"What did you learn by doing this homework?"*
3. Time you spent in:
   Thinking; Programming; Testing, Documenting; Total
4. *"What do you wish you knew before starting the homework set?"*
5. *"What would you like to explore further?"*
6. Did you collaborate? If you collaborated with another student, was that helpful?

The first question allows selecting the level of difficulty from five choices (Very easy; Easy; It was okay; Difficult; Very difficult). The questions about time spent (numbered 3 above) allow selecting among several time period options including "Other."

The three quoted questions (numbered 2, 4, 5) are designed to help students set goals and reflect on their own learning progress, which is the essence of the meta-cognitive approach we are employing. In particular, each student's answers to question number 2 ("What did you learn by doing this homework?") document her progress in learning throughout the semester. This is important since, at the end of the semester, all the experience is clumped into a fuzzy memory of topics, and one forgets what challenges one had encountered on the second or third week of the course.

Question number 4 ("What you wish you knew before starting the homework?") was intended to put a context in their learning, by revealing what they could have done better before starting. We hoped that this encourages them to consider doing so for the next assignment. Question number 5 ("What would you like to explore further?") was providing a data point for their changing ambitions during the semester.

Since one of our goals was to encourage collaboration among students, we included the last questions (numbered 6 above) to gather their thoughts on collaboration as the semester progressed.

We shared the answers to some of the questions with the whole class. In particular, the day the homework was due, we would display the graphically summarized responses for questions 1, 3 and 6. Students were eager to know how difficult the rest of the class found the homework and how much time others spent on it. We were also eager to share with the class the overwhelmingly positive comments from the students who collaborated. We observed that while there was not much collaboration recorded during the first homework, this changed in the following weeks as the level of difficulty increased and the comments from those collaborating revealed a largely positive experience.

We used Google Documents (`http://docs.google.com`) to implement our questionnaires and process the answers. This tool makes it easy to create new questionnaires, share them, summarize the answers, collect them on spreadsheets and present them graphically. A sample questionnaire can be found at [5].

Towards the end of the semester, and before the beginning of the final project, we collected into a document and sent each student her own responses from all the questionnaires during the semester. We did so by personal email. This is something we had announced at the beginning of the semester so students were expecting it. We believe sharing this document[3] is an important part of our instrument. In this document, each student can review her progress and reflect on her semester-long learning in the course. It also offers her a reminder, in her own words, about where she was standing at the beginning of the course and compare it to where she is now. Lastly, it offers a reminder of her collaboration experiences just in time as she chooses a project partner (notice that working in pairs in the final project is a requirement.)

As we mentioned, we also had a shorter version of the questionnaire which we used together with each exam. Since no collaboration was permitted in the exams, the last question (experience on collaboration) was not included, while a few others were adapted. In addition to the time it took to complete the exam, and its perceived level of difficulty, we were asking about their perceived level of fairness of the exam (since exams are primarily designed to evaluate their knowledge). We were not asking them to set goals, as in the homework questionnaires, but we were asking them about what they needed to review or learn on their own, giving them (and us) an indication on how closely they followed the course. Below are the questions of the sample questionnaire (also available at [5]):

---

1. How many hours did you spend working on this exam?
2. *"What did you learn by doing this exam?"*
3. How fair do you consider this exam to be?
4. How easy/difficult do you consider this exam to be?
5. *"Did you have to refresh your memory or learn something new in order to do this exam? If so, what?"*

The results of the exam questionnaires were also included in the document we sent to the students, mainly for completeness of their learning record.

We estimate that it takes about half hour to create the questionnaires, in the beginning of the semester. It also takes a

[3]The mailing system we used indicates that all of the students opened those emails (though we do not have a way of verifying that all of them actually read it and in what detail).

few minutes per student to create the documents with the answers and send the emails towards the end of the semester. Overall, the instructor's time commitment for implementing this instrument is minimal. On the students part, the time commitment is not big either. Our students reported that it took them less than five minutes to complete each of the questionnaires, while our records indicate that 95% of all questionnaires were indeed completed.

3.3 Sample Answers

Below we give just a few sample responses given by the students. We do not give sample answers to questions "What did you learn by doing this homework?" (they were reciting the main concepts that the homework was covering). The answers below were selected mostly at random.

- *"What do you wish you knew before starting the homework set?"*

  How do I increase the space for a program? it kept crashing when I tried to make larger trees.

  I wish I knew that JAVA returned negative numbers when modding negative numbers. I especially wish I knew ctrl-D meant null rather than "" because that wasted 2 hours of my time.

  I wish that I knew to be so careful with my text files when I am reading them in. I also wish that I knew that file IO can be so time-consuming!

- *"What would you like to explore further?"*

  Please go over all the different sort algorithms already!

  More elegant ways to input larger amounts of data.

  How to check if something is in the correct URL format in a file! It didn't work with the try catch on malformed url.

  I'd like to study iterators much more closely so I can understand them thoroughly.

- *"Did you collaborate? If you collaborated with another student, was that helpful?"*

  Yes!!! We helped each other out a lot. Even when we both had different approaches for something, we realized that what we were both trying to do was actually the same thing, we were just using different words. When I got stuck in one part, she usually had a solution for it and vice verse. I think this homework was done a lot quicker than if I had done it by myself.

  I did not work as a partner with someone, but I discussed my ideas and issues with my program with other students, which was really helpful.

  Well, it gave me incentive to not just throw my arms up in the air and say "ohwelltoobadmylifesucks". That incentive came in the material form of kicks in the shin. Ouch. 0_o

  When you get tired of looking at the code, your partner may come up with fresh perspectives.

- *Comments about the final project*

  I've learned that communication is very important. It's helpful for you and your partner to tell each other what you're working on so you're not both doing the same work, or so that everything is getting done. Another aspect of communication is being able to communicate the work that you've done so that your partner can understand it. If you can do this, then it's easier for your partner to jump in and pick up where you left off, or provide constructive feedback. I realized after testing one program for a long time that it's important to take the time to explain your thought process. I also learned that it's very useful to step away from the computer to think about the problem. When you're near the computer, there is great temptation to press compile and run for every little bit of code you edit, and that is not very efficient.

  I discovered that developing and writing classes from scratch for a self-defined purpose required different planning skills when compared to writing methods for specific purposes defined in a homework assignment. I felt that this was probably the most valuable lesson of the project.

  I loved working on the project. The fact that I had a partner helped me greatly and broadened the range of ideas for the project I had. I'm really proud of what we managed to create by the end, aware that we couldn't have done it if we hadn't been a team and slightly amazed at some things we ended up implementing. I had fun and become a better programmer as a result.

4. CONCLUSIONS

In this paper we describe a system designed to help students learn through reflection in a CS-2 programming course, though it can be easily adapted in other programming courses. The development of our instrument was inspired by educational research indicating that student reflection has positive effect on their learning by setting learning goals and monitoring them. Even though it is the student's job to do so, an instructor can facilitate the process by utilizing an instrument, like the one we describe here.

Our instrument consists of a sequence of questionnaires associated with each homework set (there were 8) and each midterm exam (there were 3). In addition to the usual questions that inform the teacher about the perceived level of difficulty and time spent on each assignment, the questionnaires include questions which encourage students to reflect on their learning, set their own goals and monitor their progress towards achieving them, over the course of the semester.

An important aspect of our instrument is that the students receive a record of their own responses towards the end the semester. This helps them realize their progress which results in increased confidence in their programming abilities.

One important characteristic of our system is that it can be implemented and set up with relative ease, making it possible to be adopted by other instructors. It is also "scalable" as it can be used in large classes without significant increase in teacher effort.

In our instrument we also included specific questions about collaboration, showing how one can augment the instrument

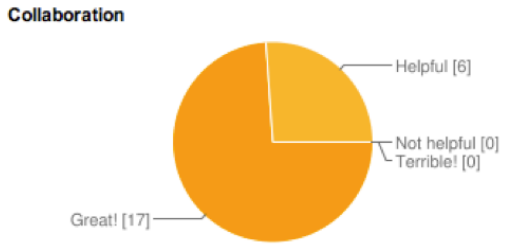

Figure 3: Student responses on how helpful they found collaboration with a partner for the final project.

so that it achieves some extra objectives without losing focus. In the final questionnaire, three fourths of the students indicated that they had a great experience collaborating in the final project (See Figure 3) while no one found collaboration less than helpful.

We believe that student reflection is a very powerful though underutilized educational principle, and we hope that other faculty will make use of our system. In particular, it would be very interesting to have a controlled evaluation of our system in a two-section class.

5. REFERENCES

[1] SIGCSE publications. http://www.sigcse.org/resources/publications.

[2] J. Allert. Learning style and factors contributing to success in an introductory computer science course. *Advanced Learning Technologies, IEEE International Conference on*, 0:385–389, 2004.

[3] A. Fekete, J. Kay, J. Kingston, and K. Wimalaratne. Supporting reflection in introductory computer science. *SIGCSE Bull.*, 32:144–148, March 2000.

[4] C. for Studies on Behavior and Development. *How Students Learn: History, Mathematics, and Science in the Classroom*. National Research Council, http://bit.ly/prH62Y, 2005.

[5] S. Kakavouli and P. T. Metaxas. Sample homework and exam assessment form (reflective questionnaire). http://bit.ly/nlMVrV and http://bit.ly/oHCABc, Last retrieved on Aug. 25, 2011.

[6] J. Kay, L. Li, and A. Fekete. Learner reflection in student self-assessment. In *Proceedings of the ninth Australasian conference on Computing education - Volume 66*, ACE '07, pages 89–95, Darlinghurst, Australia, Australia, 2007. Australian Computer Society, Inc.

[7] M. Lawson. *Being executive about metacognition*. Cognitive Strategies and Educational Performance, E. Kirby (ed), 1984.

[8] L. Thomas, M. Ratcliffe, J. Woodbury, and E. Jarman. Learning styles and performance in the introductory programming sequence. *SIGCSE Bull.*, 34:33–37, February 2002.

[9] T. VanDeGrift, T. Caruso, N. Hill, and B. Simon. Experience report: getting novice programmers to think about improving their software development process. In *Proceedings of the 42nd ACM technical symposium on Computer science education*, SIGCSE '11, pages 493–498, New York, NY, USA, 2011. ACM.

[10] L. A. Williams and R. R. Kessler. All i really need to know about pair programming i learned in kindergarten. *Commun. ACM*, 43:108–114, May 2000.