Technology Enhanced Learning for Programming Courses – Experiences and Comparison

Mirjana Ivanović, Živana Komlenov
Department of Mathematics and Informatics
Faculty of Sciences, University of Novi Sad
Novi Sad, Serbia
{mira, komlenov}@dmi.uns.ac.rs

Stelios Xinogalos
Department of Technology Management
University of Macedonia
Naoussa, Greece
stelios@uom.gr

Abstract—Methods and approaches behind technology enhanced learning (TEL) in programming courses at a university level encourage continuous research in the last 20 years. Still there is no generally applicable way that would guarantee success. In this paper some experiences gathered during years of a technology-enhanced approach in teaching programming at two universities in two countries are presented and compared.

Keywords-e-learning; object-oriented programming; teaching programming; technology-enhanced learning

I. INTRODUCTION

Pedagogical theories and methodologies underpinning technology-enhanced learning in introductory programming courses attract teachers’ community and foster continuous research since the end of the nineties. However, it still has not led to a generally applicable way of teaching and learning that would guarantee the best possible success. This inspired us to analyze, present, and compare selected experiences gathered during nearly a decade of technology-enhanced approach in teaching object-oriented programming languages (OOP) at university level, with specific focus on Java.

The analysis of the approaches and technological tools employed was conducted at two universities in two countries (Serbia and Greece). Our motivation is to try to give recommendations for delivering an introductory course on OOP using TEL. Our institutions apply some of the existing methods and use a set of suitable software tools to enhance everyday teaching practice.

Within CS1 course, at the University of Novi Sad, Faculty of Sciences (UNL-PMF) Modula-2 is used, and at the Technology Management Department at the University of Macedonia (UOM-TMD) the programming language C is used. Imperative first approach is adopted at both institutions, and OOP is taught within a subsequent course based on Java, a programming language that is widely accepted as a good language and platform for an introductory course on OOP [1].

We have examined its learning design and quality at both institutions, in order to extract recommendations for planning, designing, deployment, and evaluation of similar courses.

In the following sections we present information regarding the OOP course profile and the application of TEL at UNL-PMF and UOM-TMD. Specifically, in section II we present the aims, the teaching approach utilized and the learning design of the course at both institutions. In section III, we present how both institutions apply TEL at their OOP courses. Finally, some conclusions are drawn based on the experiences and the comparison of the applied approaches at our institutions.

II. COURSE PROFILE

At both UNS-PMF and UOM-TMD it is required that second year bachelor students master essentials of Java and become able to use fundamental concepts of OOP. Therefore, it was not difficult to identify common goals of our courses [2]:

- focus on fundamental OO software development tasks and programming concepts rather than simply learning Java constructs;
- comprehending and using standard library classes;
- analyzing/ extending existing user-defined classes;
- designing simple OO applications;
- implementing programs in Java.

Students willing to cope with advanced Java concepts, motivated by the requirements from business [3], can further upgrade their Java programming knowledge in subsequent courses offered at their university or by self-studying [1].

A. Teaching Methodology

Teaching methodologies applied within programming courses depend on a range of factors: students’ motivation, good balance between theoretical and practical aspects of teaching, tradition of teaching at a particular university, teacher’s specific teaching style, etc. Even though the university community adopted the object-orientation as an appropriate paradigm with strong expectations, it appears that it is not a particularly easy task [4] and needs a lot of effort to adjust the teaching style and methodology appropriately.

The traditional method of teaching OOP at UNS-PMF was to use the typical face-to-face lectures together with assigning students programming problems to solve. This method, however, worked well only for good students with high analytical problem-solving skills. Since UNS-PMF believes that programming skills should be acquired in interaction with
other people and from a wide variety of sources, its current practice applies blended learning modus by supporting the traditional course with online tools for delivery of self-study instructional units, assignments, discussions, examinations, and other pedagogical aids. Software solutions that are used to support such a delivery of blended OOP course are:

- learning management system (LMS) Moodle [5] for course organization, (adaptive) delivery of additional resources [6], and a variety of communication, collaboration and testing facilities,
- custom-made Web-based tutoring system within the integrated learning environment named MILE [7], which provides additional resources and provides high interactivity, offering many examples and exercises,
- IDEs, namely BlueJ [8] and a little bit further in the semester Eclipse [9], for presenting the main concepts of OO design and programming during theoretical exercises and students’ self-practice,
- code visualization tools like Jeliot [10] for providing concrete representation of the dynamic aspects of presented programs and improving students’ attention,
- in-house submission system called Svetovid [11], for efficient collection of students’ solutions to practical assignments and their timely and efficient grading.

The course at UOM-TMD is based on the microworld approach to teaching programming [12] and the educational IDE BlueJ [8]. First, a brief (2 weeks) introduction to OOP concepts takes place based on the microworld objectKarel [13] with the aim of familiarizing students with the most fundamental concepts in a clear and intuitive way. objectKarel is based on Karel++ [14], the well-known metaphor of the world of robots carrying out various tasks in a restricted world. It constitutes a learning environment that incorporates:

- a learning module with brief and concise theory and hands on activities for familiarizing students with the taught concepts before implementing them,
- a programming environment, incorporating a structure editor for developing programs, enhanced error reporting for the very few syntactical and semantical errors that can arise, program animation with immediate feedback on the depicted world of robots and explanatory visualization.

Next, the BlueJ IDE and Java are used for presenting the main concepts of OO design and programming. Students use the interactive interface of BlueJ in order to construct objects, invoke their methods and inspect their state without having to write from the very beginning a main method. However, these interactive features of BlueJ are used with caution, since their extensive usage can favor the appearance of specific difficulties and misconceptions regarding the dynamic aspects of OOP [15], [16]. The Jeliot extension for BlueJ is also used for supporting students in comprehending the dynamics of OO programs. In the middle of the course the professional IDE JCreator is also presented and students are left free to decide on their own which environment fits better to their needs.

Overall, the adopted approach is an “objects-first” (within the course), iterative (important concepts are taught first and often), and project-driven approach. The use of the technology-enhanced environments objectKarel and BlueJ plays an important role in applying this teaching approach. Both experience and the results of the long-term evaluation of the course have shown that the combined use of these environments has positive results [17].

As is the case for UNS-PMF, UOM-TMD also uses an LMS, called CoMPUs, which is mainly applied as an asynchronous e-learning platform that was implemented for supporting course management for all the departments at the University of Macedonia. CoMPUs is used for: organizing and delivering the educational material and additional resources to students, collaboration and communication, assigning projects to students and collecting their assignments.

It can be concluded that both institutions are convinced that programming skills should be best acquired in interaction although some aspects of the actual pedagogical methodology at employed software tools may differ. UNS-PMF practices blended-learning modus by offering a wealth of (adaptive) self-study material to students, together with modern and efficient, semi-automated assessment of their work performed in lab using specialized custom tools. UOM-TMD focuses explicitly on the proven benefits brought by the microworld approach to teaching programming and the chosen set of tools, together with practicing a project-driven approach within lectures, labs and homework assignments as well.

B. Learning Design

At UNS-PMF the course consists of 2 hours per week of lectures, 2 hours per week of theoretical exercises and additional 2 hours per week of lab exercises (during the semester that lasts for 12-13 weeks). In the lectures the teacher explains crucial OO concepts using PowerPoint slides and excerpts of code implementing the concepts presented, while within theoretical exercises students are confronted with complete Java solutions for different problems, as illustrations of the theoretical concepts acquired in the previous lectures.

Within the following lab exercises practical assignments are solved individually, ranging from very simple and straightforward ones in the first couple of weeks, to rather complex ones at the end of semester. Student’s effort is analyzed and graded on weekly basis, so these solutions form one part of their final grades (30%) in small steps. A special custom environment Svetovid [11] is used to provide students with simple mechanisms for editing and testing their code, but also to prevent cheating and allow teachers to grade all solutions promptly and effectively. The final grades are based on max. 30 points for solving practical assignments, and max. 30 points collected in three interim theoretical tests that focus on testing students’ problem-solving skills using the newly gained knowledge. Students are required to gather at least 30 out of a total of 60 points to approach the final oral exam (worth additional 40 points). Homework is given only from time to time, and is not graded, since the focus is on regular evaluation of the work performed in lab environment, interim tests and final exam.
At UOM-TMD the course consists of 2 hours per week of lectures and 2 hours per week of compulsory lab exercises (during a 13-week long semester). In the lectures BlueJ’s projects and PowerPoint presentations are used. Specifically, each lecture starts withposing a specific real-world problem that has to be modeled with an object-oriented program. A brief discussion takes place in class, in order to identify the classes needed for modeling the system. First, a simplified UML class diagram is presented and then the OO concepts which are used as basis for the implementation of the underlying classes. In several cases students are given excerpts of code implementing the concepts presented, or even brief tests for evaluating their understanding of the presented concepts. In the lab, students solve assignments, with or without the teacher’s guidance. Those assignments are sometimes submitted before leaving the lab through the asynchronous e-learning platform CoMPUs. Furthermore, within each lab exercise students are assigned homework that has to be submitted within one week.

Students’ final grade consists of: 20 points collected in the lab and by solving homework programming assignments; 20 points from middle-term, open-book exams on paper; 60 points from final-term exams. Emphasis is given on lab and homework programming assignments that require several hours of work during the whole semester. The fact that the points granted for these programming projects constitute a small percentage of the final grade is due to the lack of an automated mechanism for preventing cheating. However, experience has shown and students have knowledge of the fact that devoting time to the programming assignments guarantees success in both exams. The middle-term exam has the aim of assessing students’ knowledge, while giving both the teacher and the students the chance to acknowledge difficulties and misconceptions and take actions for tackling them in time.

III. TECHNOLOGY-ENHANCED LEARNING

TEL can have a considerable influence on the improvement of students’ attitudes towards learning, enhancing their success, increasing communication among students and teachers, and giving them confidence to study advanced subjects without pressure [18]. Recognizing these advantages in teaching programming, both institutions apply blended learning style in their OOP courses, though using different educational tools.

UNS-PMF identified the important goals that should be met when conducting a blended programming course in content organization and presentation, including basic teaching material and additional self-study resources, as well as in providing efficient communication and evaluation facilities. The course conductors at this institution agree with current experiences of other universities that programming languages can be successfully taught in Web-based environments and LMSs. Therefore, LMS Moodle [5] is used for basic course organization and presentation of study material, consisting of traditional static teaching material, adaptive eLessons, and a mix of various synchronous and asynchronous activities such as quizzes, glossaries, wikis, and discussion fora. Besides that, some of the resources used during lab exercises and assignments are formulated and graded online, but solved individually during regular classes. For self-studying purposes two possibilities are offered: using eLessons developed in Moodle, extended with basic personalization features [6], and/or using Mag, a custom Web-based tutoring system which is a part of the integrated learning environment MILE [7] that supports teaching, learning and student assessment.

The course can be characterized as learner-centered since, although presenting equivalents to face-to-face lectures, adaptive eLessons implemented in Moodle offer students a possibility to take as much time as they need to explore the available content. They can explicitly choose different paths or can be directed to different parts of the instructional material depending on their answers to the encountered questions, i.e. their previous and newly acquired knowledge. Mag is as well intended to be used by students who need additional explanations of basic OO and Java concepts presented in a more relaxed way. It also offers many simple examples and elements of scaffolding teaching and visualization that help students in understanding and adopting difficult concepts.

For testing students’ knowledge two different mechanisms are applied: Moodle’s Quiz module for testing theoretical knowledge and problem-solving skills, and submission system Svetovid [11] for collecting and semi-automatic assessment of students’ code produced during lab exercises. Using Moodle, three interim quizzes are conducted during the semester. A pool of over 250 questions has been created, consisting mainly of problem-solving questions similar to those used in Sun’s Java Certification Exams. Tests are solved in the controlled lab environment, in order to prevent cheating as much as possible. Svetovid allows students to code their solutions comfortably, incorporates significant part of standard Java documentation, detects and reports understandable and informative error messages, and helps instructors to promptly and safely collect and grade student solutions.

Regarding communication and collaboration tools, UNS-PMF mainly uses adequate Moodle features (discussion fora, instant messages, integrated mailing facilities, chat sessions, wikis, and blogs). These mechanisms allow students to share ideas, help each other to solve common issues, or contact the teachers and get feedback when a piece of advice is needed.

At UOM-TMD teaching and learning is based on two distinct technology-enhanced programming environments: the microworld objectKarel and the educational IDE BlueJ. The University’s e-learning platform CoMPUs is used for course management and delivery of material. The features of the platform used mostly for supporting students in learning, are:

- **Calendar**, that is kept updated with the lessons carried out, their content, and information for the associated educational material that is available in the platform.
- **Documents**, that presents students with a structure of folders corresponding to the lectures.
- **Students’ assignments**, for accessing information about the weekly homework assignments, and submitting them in the predefined deadline.
- **Discussion forum**, organized in sections corresponding to lectures, plus one for general topics.
The features of CoMPUs utilized mostly by the teacher for course management are:

- **Description of the course**, an area that contains a description of the didactical aims and the content of the course, the available material, the software needed, students’ obligations and grading policy.

- **Students’ assignments**, used for assigning homework to students and collecting their solutions.

- **Announcements**, for sending announcements by email to all enrolled students.

In addition, with the aim of supporting students in self-studying, additional educational (SCORM-compliant) material is being prepared. This material will have the form of a course delivered through the adaptive SCORM compliant LMS ProPer [19]. Finally, an issue that has not yet been resolved is the adoption of an existing on-line tool or the implementation of a new one, for assessing students’ programs automatically and providing them with immediate feedback. Such a solution, similar to the one employed at UNS-PMF, would certainly bring lots of benefits to both students and teachers conducting lab exercises and evaluating homework assignments.

IV. CONCLUSIONS

In this paper we presented experiences of teaching OO programming in two universities, comparing the applied approaches and methodologies, teaching styles, assessment strategies and effects that they impose. TEL is routinely applied at both institutions, and feedback is positive from both teachers and students. Students are usually satisfied with the introductory OOP course, the teaching methodology and the style of grading that teachers employ. They usually point out the great value of such a blended course to anyone who would like to learn basics of OO using Java in a pleasant and interesting way. Teachers claim that, though online activities are somewhat challenging supplements to traditional teaching and learning, they can significantly help students.

However, it is clear that an institution can successfully employ a wide variety of pedagogical methodologies and tools for TEL. There still seems to be room for in-house solutions at large institutions, but open source alternatives are clearly gaining on importance, especially when extended to support adaptability and personalization [20]. Getting and reflecting on valuable feedback represents another important issue not ideally resolved yet by large-scale learning environments.

ACKNOWLEDGMENTS

This work was produced within the multilateral project “Technology Enhanced Learning and Software Engineering” in which University of Novi Sad (Faculty of Sciences), Masaryk University in Brno (Faculty of Informatics), and University of Macedonia (Department of Technology Management in Naoussa) participate since 2011. It was also partly supported by the Ministry of Education and Science of the Republic of Serbia within the project no. OI174023: “Intelligent techniques and their integration into wide-spectrum decision support”.

REFERENCES


