Teaching Agile Software Engineering Using Problem-Based Learning

Nuha H. El-Khalili, Petra University, Amman, Jordan

ABSTRACT

Many studies have reported the utilization of Problem-Based Learning (PBL) in teaching Software Engineering courses. However, these studies have different views of the effectiveness of PBL. This paper presents the design of an Advanced Software Engineering course for undergraduate Software Engineering students that uses PBL to teach them Agile software development methods—particularly Scrum. The course also aims to develop entrepreneurial skills needed for software engineering graduates to better prepare them for the software industry. The assessment process designed for this course is illustrated. The paper shows that Scrum practices correlate with the characteristics of the PBL approach, which has resulted in a successful experience of PBL as reported by students in an end of a semester survey.

Keywords: Agile Software Engineering, Course Assessment, Course Design, Entrepreneurial Skills, Problem-Based Learning, Scrum

INTRODUCTION

Most software development projects can be characterized by “wicked problems” (DeGrace & Stahl, 1990). Lack of clear and complete requirements and fast technological changes are among reasons behind this assumption. This wickedness has led to the evolution and recognition of Agile software methodology and its various approaches such as Scrum, Extreme Programming, etc. Agile software development is a group of software development methods based on iterative and incremental development. It promotes adaptive planning, a time-boxed iterative approach, and customer collaboration. It also encourages rapid and flexible response to change (Ambler, 2012). A recent survey conducted in 2011 about the state of Agile development shows that 80% of the respondents work in organizations that adopt Agile practices. Furthermore, 66% of those organizations use Scrum or Scrum variations (Version One, 2011). Thus, teaching Agile development methodology to software engineering students is essential.

Furthermore, teaching software engineering students the necessary skills needed for work in the software development industry does not only require the acquisition of theoretical knowledge and technical skills, but it also necessitates other entrepreneurial skills that have been recognized by the National Academy of Engineering in its book “The Engineer of
These skills are: strong analytical skills, ingenuity, creativity, communication, lifelong learning, agility and flexibility, leadership, business and management and high ethical standards (National Academy of Engineering, 2004).

Transferring these skills to students is not possible with traditional lecturing teaching techniques. Problem-based learning has been acknowledged as a learning approach that is suitable for transferring such skills in ill-structured, multidisciplinary domains such as Software Engineering (Jonassen & Hung, 2008).

Kolmos and Du (2008) had evidence that PBL is effective in achieving the following qualities:

- Developing management, collaboration and communication skills;
- Developing criticality of learners;
- Promoting deep learning;
- Developing professional identity and responsibility development.

This paper presents the design of an Advanced Software Engineering course that has two main objectives: (1) the practice of software development using Scrum, which is one of the Agile development methods, and (2) the development of entrepreneurial skills needed for software engineers. The next section will report other work that has used PBL for teaching software engineering courses. Afterwards we will present the details of the course design. Then we discuss the correlation between Scrum practices and PBL characteristics will be illustrated. Followed by the assessment methods used in the course will be presented. Finally, we then will report the results of implementing the course and present the students’ feedback.

**Background**

Problem-based learning is a constructivist learning paradigm where small groups of students engage in cooperative learning and collaboration to solve complex problems in an authentic project context (Brodie, Zhou, & Gibbons, 2008). This approach has been utilized by many disciplines; first by the medical field, later by health sciences, and the fields of mathematics, business, and engineering.

Similarly in Software Engineering and Computer Science fields, many studies have reported the utilization of PBL in teaching a variety of courses. For example, Bednarik reported using this method for teaching the theoretical foundations of computer science. The course covered topics of theory of computation such as finite automata, context-free grammar and Turing machines (Bednarik, 2004). O’Kelly and Gibson use PBL, combined with a competition setting, to teach first year students programming. Students use RoboCode, which is a java programming game and a development environment to enhance and improve a game, while learning more advanced features of Java (O’Kelly & Gibson, 2006). Richardson and Delaney used problem-based learning to teach Information Systems undergraduate students how to design and model information flows in businesses (Richardson & Delaney, 2009). Meanwhile, others report using PBL to teach Software Engineering courses for Master students (M.Sc.) (Richardson et al., 2011; Cavalcanti, Santos et al., 2008). Armarego reported the utilization of problem-based learning in an advanced software design course for fourth year software engineering students (Armarego, 2002). Of particular interest to this research, two studies have reported the utilization of PBL in teaching students Agile Software Engineering. The first describes the successful utilization of PBL and Unified Process model to teach undergraduate students object-oriented software development (Qiu, & Chen, 2010). In the second, Murphy describes the challenges that arise from teaching a distance-learning advanced software engineering course that uses PBL to teach students Extreme Programming practices and principles (Murphy, Phung, & Kaiser, 2008).

However, the studies reported in literature have different views of the effectiveness of using PBL in teaching Software Engineering courses. In Mitchell and Delaney (2004), Mitchell and Delaney report that according to their experience in implementing PBL, their results show an increase in students’ Software Engineering
Related Content

Online Synchronous vs. Asynchronous Software Training Through the Behavioral Modeling Approach: A Longitudinal Field Experiment
[www.irma-international.org/article/online-synchronous-asynchronous-software-training/1692/](www.irma-international.org/article/online-synchronous-asynchronous-software-training/1692/)

Increasing the IT Knowledge of Indiana High Schools
[www.irma-international.org/article/increasing-knowledge-indiana-high-schools/2297/](www.irma-international.org/article/increasing-knowledge-indiana-high-schools/2297/)

Technology Integration and Innovation during Reflective Teaching
[www.irma-international.org/article/technology-integration-and-innovation-during-reflective-teaching/146865/](www.irma-international.org/article/technology-integration-and-innovation-during-reflective-teaching/146865/)

A Successful Failure to Collaborate on Storage Technology Education
[www.irma-international.org/article/successful-failure-collaborate-storage-technology/37520/](www.irma-international.org/article/successful-failure-collaborate-storage-technology/37520/)

Revealing Student Blogging Activities Using RSS Feeds and LMS Logs
Michael Derntl (2010). *International Journal of Distance Education Technologies* (pp. 16-30).
[www.irma-international.org/article/revealing-student-blogging-activities-using/45142/](www.irma-international.org/article/revealing-student-blogging-activities-using/45142/)