Chapter X
Experiences in Project-Based Software Engineering: What Works, What Doesn't

Steven A. Demurjian
University of Connecticut, USA

Donald M. Needham
United States Naval Academy, USA

ABSTRACT

Project-based capstone software engineering courses are a norm in many computer science (CS) and computer science & engineering (CS&E) accredited programs. Such cap-stone design courses offer an excellent vehicle for educational outcomes assessment to support the continuous improvement process required for accreditation. A project-based software engineering capstone course near the end of a student's program can span the majority of CS and CS&E program objectives, providing a significant means to assess at-tainment of these objectives in a single course location. One objective of this chapter is to explore the role of a project-based, software engineering course in accreditation. An addi-tional objective is to relate over twelve combined years of experience in teaching such a course, and in the process, highlight what works and what does not. We candidly examine both the successes and the failures that we have encountered over the years, and provide a roadmap for other instructors and departments seeking to institute such courses.

INTRODUCTION

Since its early roots at the 1968 NATO conference in Garmisch Germany (Naur, Randell & Buxton, 1976), the software engineering discipline as has sought to use tools, techniques, and paradigms similar to those found in other engineering disciplines in order to improve the quality and reduce the cost of software development. The seminal "No Silver Bullet" article by Brooks (1987) in part focuses on identifying the essence of what makes software development difficult and stresses that the ability to modify software so as to accommodate evolving hardware requirements is one

of the key aspects of understanding the inherent difficulties faced by software developers. Agile, lightweight methodologies such as Extreme Programming (Beck, 1999) emphasize customer involvement and promote team work in an effort to make the development process better suited to adapt to changing requirements. Of note with Beck's approach is the use of ad-hoc teams to resolve difficulties that arise during the development process. Software engineering educators have responded to these needs in part with the emergence of project-based software engineering capstone courses at the undergraduate level.

Such software engineering capstone courses are becoming a cornerstone of many computer science (CS) and computer science & engineering (CS&E) programs, and provide a means for practice-based exploration of large-scale projects in a team setting following current trends in software engineering course sequence design (Abran & Moore, 2004; Boehm, Kaiser & Port, 2000; LeBlanc & Sobel, 2004; Meyer, 2001; Shaw, 2000). Our approach to project-based software engineering capstone courses allows students to apply concepts and ideas garnered throughout their undergraduate program within a capstone experience near the end of their studies. For students, such courses can provide the opportunity to control the project topic, select teammates (to a limited degree), make critical decisions, and problem solve by applying coursework knowledge and their experiences. Project topics selected by our teams have run the gambit from standalone Java applications, automatic mixing machines that use windshield wiper motors and micro-processor controlled PVC pipes run via a web interface, to embedded system controllers for autonomous underwater vehicles. In such courses, instructors can serve as the mentor or project manager, overseeing the week-to-week schedule of deadlines, and arbitrating among team members when difficulties or clashes in personalities arise.

Project-based, software engineering capstone courses can also play a vital role in terms of accreditation. ABET, known prior to 2005 as the Accreditation Board for Engineering and Technology (ABET, 2007), has assumed accreditation over CS, CS&E, information technology, software engineering, and computer engineering programs. As part of ABET accreditation, departments must identify program objectives, and detail the program outcomes for each program; CS&E has outcomes that are influenced by engineering accreditation requirements, while CS outcomes have been influenced by computing accreditation requirements. Given a set of program outcomes, in order to support a continuous improvement process, it is necessary for departments to assess their programs on a regular basis. Since well-focused project-based software engineering courses can span nearly the entire curriculum in terms of topic coverage, they can serve as an ideal vehicle to accomplish this objective.

This chapter focuses primarily on CSE293, Capstone Project-Based Laboratory, in the Department of Computer Science & Engineering at the University of Connecticut (UConn). Experiences gained from a similar course, IC480, Computer Science Capstone, offered at the United States Naval Academy (USNA) are interleaved where they provide significant complementary or contrary perspectives. UConn's CSE293 was a new course developed during the Spring 2001 semester which we designed and instituted as part of our major curriculum changes for ABET 2000 accreditation. The course has been taught every semester at UConn since that time, with multiple sections by multiple instructors. The course philosophy of CSE293 is for the students (typically seniors near the end of their programs) to demonstrate the ability to work in a team with minimal or no guidance, where the team organizes, plans, designs, prototypes, and delivers a product according to milestones established (and known in advance) for the semester. Throughout the semester, the instructor delivers appropriate feedback to students in various mediums (oral, email, annotated documents, etc.), in response to