Incorporating Development for Mobile Devices When Teaching Software Engineering

Ani Nahapetian
Computer Science Department
California State University, Northridge (CSUN)
ani@csun.edu

Abstract

This work presents a case study of incorporating projects on software and middleware development for mobile devices, into existing software engineering courses of a Computer Science curriculum. As a predominant technology among our student bodies (with smart phones and tablets), along with the growing market for Android or iPhone developers, students are yearning for opportunities to develop apps and interface with mobile devices. Here, we incorporate knowledge and practical experience with mobile systems (including app development, interfacing with research systems and off-the-shelf embedded devices) with existing software engineering courses.

Keywords: Software Engineering; Mobile Systems; Body Area Networks

I. INTRODUCTION

In our case study, an existing undergraduate level software engineering course has been modified by integrating cutting-edge mobile system development with the fundamentals of software engineering. In the course, students work in teams to produce large software projects that address real-world problems and needs, while carrying out development for various mobile systems and devices. The application of software engineering practices presented in lecture, such as software versioning and unit testing, to the mobile systems course projects highlights the relevance and applicability of the software engineering practices, while also providing an opportunity for students to increase their experience working with mobile devices.

Mobile system development courses for undergraduate students are fairly limited, especially when compared to the availability of introduction to software engineering courses. Instead of proposing and developing a new course (although it has successfully been carried out), we infused mobile development into an existing required Computer Science course. The aim of this effort was to expose students to a popular industry field, without imposing any additional graduation requirements.

Throughout the course, students engaged in discovery learning, by asking questions to understand functional and domain requirements, by refining ideas during the drawing of straw man and formal designs, and by formalizing concepts during software creation and testing. The process of discovering and determining information through hands-on work such a prototype development, as well as from posing technical questions in the requirements engineering process, has the potential to clarify ideas and motivate student learning in the area of mobile computing. The advantages of using discovery learning in Computer Science curricula have been published widely [5][1][4][6], along with the significance of team-based project work in undergraduate software engineering courses [2][3][7].

II. COURSE OVERVIEW

Mobile computing topic infusion presented in this paper was carried out California State University, Northridge (CSUN)' Comp 380/L: Software Engineering undergraduate course. This course has both a lecture and a lab component, with software engineering fundamentals and best practices presented during the 50-minute lecture and the 75-minute lab time providing students with an opportunity to work and collaborate on project tasks under the guidance of the course instructor. The semester-long course involves several small teams of students (~4-5 students/team) each developing code, documentation, and configuration information for a large software project.

As a required part of the undergraduate Computer Science degree program, COMP 380 experiences a large enrollment of undergraduate Computer Science majors, typically in their junior year of college. Additionally, the course fulfills requirements for the Computer Engineering, Computer Information Technology, and Information Systems majors, and thus enrollment by students from these majors is also common.

Throughout the software project development, students work on various builds to complete their projects. Although each team project varies, various compulsory components are involved in the project work. Students work towards successive builds. By the end of the semester, the students have practiced and learned various important skills in software development for mobile devices. Additionally, they have still worked towards an understanding of software engineering principles.

Figure 1 provides the overview of the course activities through the semester. Throughout the course, students interface with a 'customer,' an invited stakeholder with an interest in the project's product. Additionally, they interact with the instructor formally with lecture and informally with the lab component of the class.

978-1-4673-5112-6/13/$31.00 ©2013 IEEE

<table>
<tr><td><b>Students</b></td><td colspan="3">Project proposal and team formation</td><td colspan="3" rowspan="2">Requirements engineering</td><td colspan="2">Software design</td><td colspan="5">Implementation and testing</td><td colspan="2" rowspan="2">Final presentation
Acceptance testing</td></tr>
<tr><td><b>Customer</b></td><td colspan="3">Background presentation</td><td colspan="7"></td></tr>
<tr><td><b>Course Instructor</b></td><td colspan="15">Presentation of software engineering topics; overseeing project progress; acting as liaison between students and customer</td></tr>
<tr><td><b>Week</b></td><td><b>1</b></td><td><b>2</b></td><td><b>3</b></td><td><b>4</b></td><td><b>5</b></td><td><b>6</b></td><td><b>7</b></td><td><b>8</b></td><td><b>9</b></td><td><b>10</b></td><td><b>11</b></td><td><b>12</b></td><td><b>13</b></td><td><b>14</b></td><td><b>15</b></td></tr>
</table>

Figure 1. Course activities during 15-week semester, for all project participants.

III. Case Study Evidence

The successfully created student projects provide evidence of the successful infusion of mobile computing topics in the lab component of the software engineering course. Select projects are highlighted in this section, including the CSUN Parking App (CPA) and WeMet.

All of the projects were implemented using the Anrdoid platform. Students in the program were exposed to Java programming, thus making it easier to work with Android API than the objective-C based iPhone platform.

The CSUN Parking App (CPA) project, developed in Spring 2012, involved the development of a participatory sensing based system for directing drivers to open campus parking lots. Figure 2 provides high-level use-case diagram for the system.

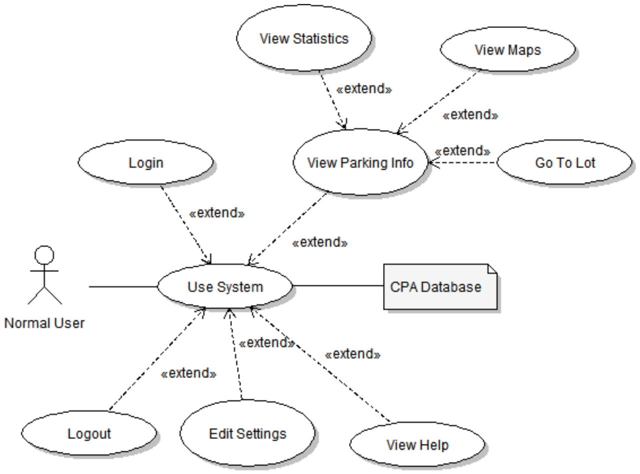

Figure 2. High-level use-case diagram for student CPS parking app project.

The project involved collecting data from an interactive form and map developed by the students. The data was sent to a remote server, which used a Model-View-Controller architecture to store and provide parking guidance for users.

The WeMet project, developed in Fall 2012, provided an add-on to a phone's contact list. It enabled the recording of the location where a contact was added to the contact list. The Google Maps API was used to visualize the locations where contacts were 'met.' Additionally, photos of the contacts could be added, to better solve the problem of forgetting the people who have been added to one's contact list. Screen shots of the WeMet project are presented with the software architecture in Figure 3.

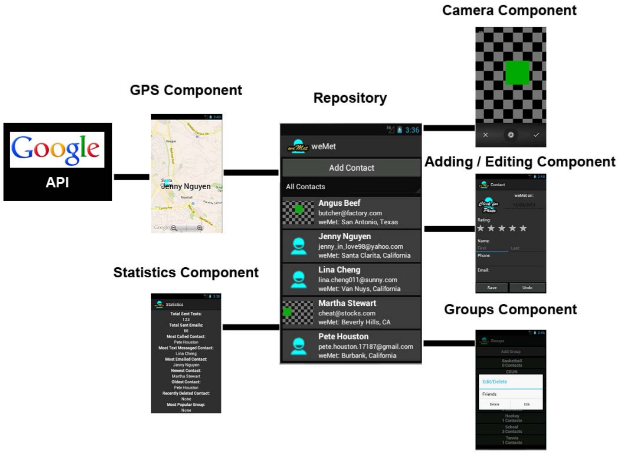

Figure 3. WeMet student project software architecture with screen shots.

References

[1] Baldwin, D. Discovery learning in computer science. (1996) SIGCSE Bull. 28, 1, 222-226.

[2] Chamillard, A.T. and Braun, K. A.. 2002. The software engineering capstone: structure and tradeoffs, (February 2002) SIGCSE Bull. 34, 1, 227-231.

[3] Nahapetian, A. Bridging Classroom Heterogeneity: A Software Engineering Course and Projects, (2008) Informatics Education Europe (IEE).

[4] Olagunju, A. O., The role of scientific discovery in teaching and learning of computer science, (December 2000) *SIGCSE Bull.* 32, 4, 28-31.

[5] Paxton. J.T., A novel approach to teaching artificial intelligence, (March 1995) *SIGCSE Bull.* 27, 1, 283-286.

[6] John A. Trono. 2008. A discovery-based capstone experience, (April 2008) *J. Comput. Sci. Coll.* 23, 4, 195-200.

[7] van der Hoek, A., Kay, D.G., and Richardson, D.J. Informatics: A Novel, Contextualized Approach to Software Engineering Education, P. Inverardi and M. Jazayeri (Eds), Software Engineering Education in the Modern Age: Challenges and Possibilities, PostProceedings of ICSE '05 Education and Training Track, Lecture Notes in Computer Science 4309, Springer, November 2006, pages 147–165.