SMARTPHONE APPLICATIONS AS SOFTWARE ENGINEERING PROJECTS

Catherine Stringfellow, Divya Mule
Midwestern State University
Wichita Falls, TX 76308
+1 (940) 397-4578
catherine.stringfellow@mwsu.edu, divya.mule@yahoo.com

ABSTRACT
This paper describes issues in developing mobile device applications and using smart phone applications as projects in a software engineering course. It also describes the experiences of the students in developing such a project.

1. INTRODUCTION
Software engineering is an important course in the computer science curriculum, but many students feel bored by the theory part in the course and by some of the projects they are asked to develop. Faculty often employ real world projects and game projects to motivate and interest students. Given the recent increase in smartphone use, especially among computer science students, more courses are being offered in smartphone development or smartphone development is being incorporated into project-based courses. This paper describes issues in using smartphone applications as projects in a software engineering course.

Section 2 describes issues involved in developing applications for mobile devices in general and the issues involved in using them in classroom projects. Section 3 describes the Android smartphone application project used in our course, while section 4 describes the project and the concepts and issues encountered. Finally section 5 presents conclusions.

Copyright © 2013 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.
2. BACKGROUND

In a software engineering course, students are exposed to the idea of software quality and software quality assurance. Some of these qualities need special attention in developing applications for smartphone devices – including security, integrity, efficiency, testability, portability and especially, usability. There are also several important issues to consider in regards to developing such applications in project based courses, like software engineering.

2.1 Issues in Mobile Application Development

Some issues in developing mobile applications include security concerns, usability for a mobile device, and development and testing tools. Lindquist, Anguamea, and Ramamurthy [9] state that Java Frameworks provides means for handling passwords, certificate keys, digital signatures and encrypted data that may help developing more secure distributed limited device applications. Myers, Nichols, Wobbrock, Miller [10] investigated usability - they made recommendations for the types of applications that should stay on a PC and those that would be better for handheld devices. Grønli, Hansen, and Ghinea [7] studied usability of mobile applications in terms of accessibility. Java ME supports applications that are useful with a text-to-speech feature and Android supports assistive environments.

Another big concern is context-awareness – a common usage requirement for smartphone applications that relates to the context of execution of a device or the ability of a system to gain and use information based on the system’s environment. Grønli et al. [7] address location aware capabilities. Gellersen, Schmidt, and Beigl [4] discuss the issue of context awareness beyond location by reporting their experiences on augmentation of mobile phones with awareness technologies and on embedding of awareness technology in everyday non-digital artifacts, such as an active badge.

Grønli et al. [7] also compare three development environments in terms of implementation and performance aspects and quality assurance. Android and Windows Mobile support a better development environment that works well with actual devices, as they are closely integrated with their operating system, while Java ME has poor emulator support, but supports many types of devices. Bo, Xiang, and Xiaopeng [1] state there are difficulties in testing applications for smart phones, due to the diversity of platforms, performance and storage capacity, and the interactive interface. So in addition to mobile application development tools, tools are required to test software embedded in mobile devices.

2.2 Issues in teaching mobile applications

Several researchers have reported on their attempts to develop applications for student use or have students develop the applications themselves in various computer science courses. Grissom [6] determined that courses on HCI, internet programming, computer ethics, graphics, networks, architecture, and operating systems teach concepts useful for designing mobile device applications, while the software engineering or
capstone courses give students experience in developing mobile applications. Incorporating the latest technologies into the curriculum, however, is challenging to both faculty and students. Mahmoud et al. [11] describe an academic kit to expose instructors and students to mobile application development. Teaching modules of mobile application development are integrated in existing core courses like software engineering, web services, game design and development, information security and operating systems, etc.

Riley [13] taught a Java course electing to incorporate the Android platform. The Android platform is based on the Java programming language, has a plug-in with Eclipse, and an emulation environment called Android Virtual Device (AVD), thereby not requiring the use of an actual mobile device for basic development. O’Rourke, MacDonald, and Goldschmidt [12] points out that since iPhone applications cannot be downloaded into an actual device using Apple’s SDK, students must test their mobile applications in the simulation environment. Simulators have drawbacks, though; for example many of them cannot simulate camera features.

Some universities are now offering courses in smartphone programming. Flynn and Poellabauer [3] describe their course that not only gives students experience in developing mobile systems applications, but also teaches them concepts of wireless communication and computing. Ivanov [8] describes an iPhone course where the application projects use location context awareness. Yan, Becker and Hecker [14] describes a course on iPhone application development that covers concepts in the underlying language (Objective C), the iPhone SDK and frameworks, useful development tools, and common design patterns in smartphone application development. Gestwicki and Ahmad [5] use App Inventor in their course to create applications for Android. App Inventor uses a visual programming language to focus on the significant behavior of programs. England [2] reports teaching an upper level CS course using Apple iPad as the target platform to address student needs for a team project, exposure to another programming language and development environment. Concepts covered include object-oriented programming, design patterns, and collaborative and software versioning.

3. PROJECT

Software engineering is an important course in the curriculum, but many students feel bored by the theory. Socially relevant computing is one way to gain interest. Another way is mobile applications. Students use smart handheld devices, but their understanding of how these devices work and how they are programmed is limited. Students are also not aware of the quality issues—usability, portability, etc., that are especially important in developing mobile smartphone applications.

Students in an semester-long senior software engineering course were grouped into three teams of four and given a choice between either a smart phone app or a stand-alone app in C#, a language with which many had experience. Every team chose the smartphone app, indicating that it was more appealing, challenging, and relevant to real world programming. The students also stated having mobile application development skills would be a useful experience for their future and something they could put on their resumes. The students were then given detailed specifications of the project and prototypes developed by students in the previous semester. The project required an app that helps friends and colleagues in deciding where to go to lunch when they cannot agree
by making the choice for them. A list of restaurants is generated based on several different criteria, such as type(s) of food desired (Mexican, Italian, Asian, etc.), style of restaurant (buffet, dine-in, drive-through) and cost. Restaurants that meet the criteria randomly populate a GUI control for a game, which then makes the decision. The students were given some freedom in selecting a game; although they were given some examples, such as a plinko game, spinning wheel, space invaders, race, etc. The application had some non-functional requirements, too, such as a nice GUI, minimizing memory and downloading time, and good fun interactivity. The application was to run on Android devices.

4. STUDENTS’ EXPERIENCE

Students were surveyed at two points in the course – after reviewing the prototypes and at the end of the course – to get their views on their experience in developing a smartphone application in a software engineering course. The students were given the assignment to look over several prototypes and critique them, then rewrite the specification documents for their own project. This activity helped them to develop their critical thinking skills and helped them recognize the value of good documentation. Figure 1 shows two screenshots from one of the prototypes with their critical comments. The students indicated the prototypes had various defects that caused the systems to crash (which is expected in a prototype). Aside from defects, there were issues with the non-functional requirements of those prototypes, such as performance and usability. There was not enough space for the displays (the game and map shared screen space) and the games were not very exciting.

Figure 2a shows a screenshot from one the final projects with a slider to choose distance. This app had three separate screens for playing the game, showing the results (and asking “deal” or “no deal”) and for displaying the map. Figure 2b shows screenshots from another project showing a more interesting game GUI involving racers and the final results screen.
At the end of the course the students were asked the questions:

*If they had more time, what would they do?*

*What could have been better?* and

*What did they learn?*

With more time and resources, they might have instead been able to manually create the database for local population areas, or at least find a more robust source of information. User reviews, either through Yelp or an application-side implementation, could also have been added. They felt they could further improve the look of the interface; in particular, animations could be added to the game screen. In answer to the second question, the students indicated the design and implementation could have been better, the game quality improved, and very specifically, they recognized they needed to remove multiple instances of restaurants in the area (e.g. McDonald’s) and reduce it to the closest franchise. Their critiques of their applications concerned the map display and Google Maps API, different ways to decide winners in their games, and adding feature, for example, a shake function so that it would cause the racers in the race app to spin out of control. Another feature would involve auto-completing input text boxes. In regards to what could better their experience in developing an Android application, students indicated they did not like the Android emulator and preferred testing the apps on their own Android smartphones. This clearly identifies the need to address testing of smartphone application tools.

Finally, in regards to what they felt they had learned, the students thought this project taught them a great deal both technically and socially. They recognized it is necessary to practice proper time management techniques and realized this is not just important in the classroom, but would be in the real world as well. Another valuable lesson was learning the extent to which documentation is generated for a single software product. Of course, they also learned a great deal about Android implementation and the Java programming language. Beyond what they learned, they appreciated the experience...
working on software with a team and having the opportunity to create a useful software tool that can be utilized effectively. Such projects are normally out of the grasp of a single software developer.

5. INSTRUCTOR’S OBSERVATIONS

Issues related to non-functional requirements, such as efficiency, portability and usability, did come up in the course of developing this project and to a greater extent than other types of projects. Students had to be concerned about the limited memory on a phone and the need for fast downloads due to limited data plans. Students had to be concerned with portability, as the students’ devices (and the instructor’s phone) had different versions of the Android operating system. The student cared most about usability, as location context awareness was necessary for the project, user inputs were mostly non-textual and the game had to be interactive and fun. Particular to an Android project assignment, students on their own furthered their knowledge of inheritance, event-driven programming, memory management, efficiency, and using APIs, including creating their own.

Students had to take the initiative in researching online documentation to learn how to use the tools, configure them, and overcome certain technical issues, such as transferring an app to an actual device (which actually is not difficult). Given the experience of teaching a course with an Android project, the following advice can be offered. Despite the students’ complaints about the Android emulator in Eclipse, it was quite useful, especially for those few remaining students who do not have their own Android device. The emulator is a bit slow, but it has improved. Java “compatibility” or AVD configuration errors are common, frequently when going back and forth working at home, the lab or using a different device, and easily overcome – a handout describing a few of these errors and explaining how to fix them should be provided. Also, it is quite easy in Eclipse to connect to an actual Android device instead – useful if students are testing apps for an Android 3.0 device or later. Having a few Android devices available for checkout was found to be helpful. (Make sure drivers are installed on lab computers for these devices.)

Students should be provided a list of free online tools, which can analyze code. There are tools which can produce UML diagrams, create documentation and provide static metrics – topics very appropriate for a software engineering course. Students should also be given a list of various Android forums to help them solve their technical issues and links to online videos for installing and configuring the tools. Table 1 has a small selection of some of these links.

Table 1. Useful links for students developing Android projects in a software engineering course.

<table>
<thead>
<tr>
<th>Topic</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Android ADK</td>
<td>developer.android.com/sdk/index.html</td>
</tr>
<tr>
<td>The ADT Plugin for Eclipse</td>
<td>developer.android.com/sdk/eclipse-adt.html</td>
</tr>
<tr>
<td>Creating AVDs</td>
<td>developer.android.com/guide/developing/devices/managing-avds.html</td>
</tr>
<tr>
<td>Beginner level help</td>
<td>stackoverflow.com/questions/tagged/android</td>
</tr>
</tbody>
</table>
6. CONCLUSION

Some of the issues identified in this paper did, indeed, arise in the development of the smartphone applications in the course. In this project, it was observed that students felt security was not a concern, but efficiency, usability, portability and context-awareness were. The development tool (Android Emulator), and the use of APIs were of most importance. Certainly, the nature of the project will dictate which issues are of concern for that project.

Students do appreciate developing applications that they know will be used or in gaining experience that they feel will serve them after graduation. In a software engineering course with mobile application projects, students learn the concepts of HCI, incremental and iterative processes, quality assurance, evaluation and collaboration. This paper described some of the challenges the students faced and demonstrates that students encounter certain concepts often covered in software engineering, but perhaps not seen to the extent that they are in developing smartphone applications.

7. REFERENCES


