USING INTERDISCIPLINARY TEAMS IN A MOBILE APPLICATION DEVELOPMENT COURSE[1]

Frank McCown and Stacy Schoen Gibson
Computer Science Department and Department of Art and Design
Harding University
Searcy, AR 72149
fmccown@harding.edu and sschoen@harding.edu

ABSTRACT

Due to the overwhelming popularity of smartphones and tablets, many computing departments are introducing mobile application development (Android, iOS, Windows Phone, etc.) courses into their curriculum. Art and Design departments are also equipping their students to design graphics for this growing market. This paper describes an experiment combining undergraduate computer science and graphic design students on interdisciplinary teams to develop Android applications. After spending the semester developing the apps, students were asked to reflect on their experiences. The responses have been analyzed and the outcomes from the experiment are being shared. It is hoped the findings in this paper will aid other instructors who are thinking about merging the talents of graphic design and computer science students to develop software.

INTRODUCTION

As smartphones and tablets become increasingly popular, computing departments are introducing mobile application development (Android, iOS, Windows Phone, etc.) courses into their curriculum [7]. These courses are quite popular because students are motivated to learn how to write programs for a device they and their peers use all the time, and the job market is very strong for mobile application developers. A significant difficulty of developing mobile apps is that many computer science (CS) students are not trained as graphic designers, so their creations may not look as polished or designed as the apps they commonly use. Meanwhile, Graphic Design departments are equipping their students to take traditional design skills and apply them to new and growing formats, such as phone and tablet design. While design students are being trained to develop aesthetically pleasing interfaces, layouts and graphics, they rarely see their design functioning in an actual app.

Combining the talents of CS students and graphic design (GD) students into interdisciplinary teams is ideal for several reasons. First, it allows the CS students to focus their efforts on writing code instead of creating layout and icons. Graphic designers are able to develop the graphical artifacts and aid in designing visually appealing interfaces and, perhaps for the first time, are able to see their artwork actually being put to use in software. By working together, students are able to produce mobile applications that are not only functional but also aesthetically pleasing. Second, the division of labor mirrors the makeup of "real world" software development where programmers and graphic designers work together to develop software [2]. Anything educators can do to expose undergraduates to real-world software development practices will ease the transition from academia to industry. And finally, interdisciplinary teamwork allows CS students to develop the "soft skills" like communication and teamwork that employees are looking for [3, 4]. CS students benefit by working with students who do not share their same working vocabulary and who are less concerned about programming details. GD

students benefit by seeing first-hand the practical implications of incorporating their design into software.

In the fall of 2011, an experiment was conducted at Harding University combining undergraduate CS and GD students into interdisciplinary teams that developed Android applications from initial concept to beta release. This paper reports details about the Android class, how the teams were created, and how the class progressed. After presenting results from the course surveys, lessons learned are presented that may be helpful to other instructors who are interested in using interdisciplinary teams of GD and CS students to produce software.

ANDROID COURSE & TEAM FORMATION

Computing departments are increasingly looking for ways to engage their students in interdisciplinary teams so as to improve their students' non-technical soft skills that employers desire in college graduates [3, 4]. Some universities have created entire courses which combine students from different disciplines and majors into interdisciplinary teams [3, 5, 6]. Others have combined students from concurrent courses in different disciplines into teams that work on the same project [1]. This second approach was used at Harding University because it allowed GD students to participate with CS students on an Android app project as part of their routine course work.

In the fall of 2011, the Computer Science Department at Harding University offered an elective course called Android Application Development for upper-level CS majors. Seventeen students (two were females) enrolled in the class. The course was modeled after the Android App Course developed by David Janzen at Cal Poly [10] and focused on the software engineering process where teams of students develop a single Android app from start to finish. There were fixed deliverables throughout the semester including horizontal and vertical prototypes, alpha and beta releases, and app evaluations. Students worked on tutorials throughout the semester to learn how to write Android apps [9].

During the same semester, the Art & Design Department offered the Advanced Graphics Design course, a required course for senior graphic design majors. Twelve students (five were females) were enrolled. The class was to participate with the Android students on interdisciplinary teams, and their work was counted as one of their major projects. For logistical reasons, the Android course and Graphic Design course did not, unfortunately, meet at the same time.

At the beginning of the semester, the CS students were divided into eight teams with two or three in each team. The themes for the Android apps were chosen by the CS students with the aid of the instructor. Themes included a wedding planner, football play design, geography game, maze game, friend-tracker, dream logging, local news and weather, and a school information app [8].

Computer Science Students	Graphic Design Students
Write all code	Develop the app icon
Integrate artwork into app	Design the splash page
Screen templates & navigation	Screen templates & navigation
Alpha and beta testing	
Turn in all deliverables	

Table 1: Division of labor between CS and GD students.

After the themes had been chosen, one or two GD students were assigned to each team, and an initial meeting was set when all CS and GD students could meet during lunch time. Team members were introduced at the meeting, and the instructors laid out their expectations for working together and identified the proper division of labor as shown in Table 1. The CS students had significantly more responsibilities than the GD students because the CS students were to work on the project over the

entire semester whereas the project was but one of several for the GD students. While the students had different roles to play, they had to work together to design the screens and envision how the user would interact with the app. Teams were encouraged to plan a few initial meetings to create mockups of user interfaces for the apps and create some initial graphics for the horizontal prototype which was due the following week. The instructors challenged the teams to develop apps that were of sufficient quality that they could be placed on the Android Market, and the students seemed very excited by the prospect.

INTERDISCIPLINARY COMMUNICATION

All the teams successfully turned in horizontal prototypes incorporating initial artwork designs from the GD students. After the prototypes and designs were evaluated by both instructors, most of the CS students turned their attention to coding, getting their apps ready for the next few deliverables: vertical prototypes and pre-alpha releases. The instructors did not require regular meetings during this time because the GD students had other projects to work on as part of their course work, and most of the work now involved coding. A few motivated CS and GD students decided to keep meeting, but most ceased communication for several weeks.

A few weeks later when the due date for the beta drew near, most teams began to get into contact more regularly because the beta had to incorporate all artwork composed by the GD students. It was at this point, however, when the instructors began to notice problems in interdisciplinary communication. Some CS students complained that their GD teammates were not responding to emails or seemed too busy to meet with them. Some GD students complained that the CS students were equally non-communicative. When the instructors became aware of problems, they would try to help rectify the situation. Students gave a number of reasons for the communication problems including personal issues, overly busy schedules, laziness, and severely introverted personalities. Both the CS instructor and GD instructor had to encourage their students on several occasions to live up to the commitments of being on a team.

Because the instructors did not require meetings or have any built-in consequences for failing to work together (besides the end result of producing a sub-par app), not all students felt as motivated to work through their problems. These problems and others were identified and elaborated on in the course survey (discussed next).

COURSE SURVEY

At the end of the semester, both CS and GD students were required to complete a course survey which probed their attitudes about the class project and their teammates. The survey included questions with Likert-type scale responses and open-ended questions. The results of the Likert-type scale responses are summarized in Table 2. Note that the two questions under “Attitudes about my Teammates” asked the CS students’ attitudes about their GD teammates and vice versa.

As can be seen from the table, a majority of CS and GD students (82% and 67%) enjoyed working on the project. The CS students reported being more challenged by the project (100% vs. 50%). Overall, the CS students were quite pleased with the graphics produced by the GD students, but the GD students themselves were less pleased with 42% of them having a neutral attitude about their work. The vast majority of CS and GD students (88% and 83%) thought the work of the GD students significantly improved the visual appeal of the project. As one CS student put it, “Without the graphic designer, my project would have looked like a DOS terminal.”

The attitudes between the CS and GD students differed the most when it came to implementing the graphics into the app. Although a majority of CS and GD students (65% and 59%) were at least somewhat pleased with how the graphics were implemented, a quarter of the GD students were displeased with the implementation. One voiced her disappointment on the survey writing, “The final app looked nothing like the designs given to my comp sci team. I'm not sure if they didn't know what

they were doing and were afraid to tell us or if they just didn't like our design and wanted to design it on their own." Another GD student was even more disheartened stating, "I spent a large amount of time designing [graphics that weren't] used in the final design. I felt like I did all this work for nothing." And another said that their CS teammates "changed our designs without asking and didn't change what we asked them to change." On the other hand, one CS student wrote that they did not receive some of the graphics they requested from their teammate, and another said, "Perhaps [the GD student] didn't have much time to put into it. The graphics were very simple, and I feel I could have come out with better graphics." These responses indicate a serious rift in a few teams that point to problems stemming from a breakdown of communication, inability to implement designs properly, unrealistic expectations, or perhaps some territorial dispute. These problems will be addressed in the next section.

Attitudes about my Project		Strongly Disagree	Somewhat Disagree	Neutral	Somewhat Agree	Strongly Agree
1. Enjoyed working on the project	CS	0%	6%	12%	29%	**53%**
	GD	8%	8%	17%	**42%**	25%
2. Project was challenging	CS	0%	0%	0%	**59%**	41%
	GD	0%	25%	25%	**50%**	0%
3. Pleased with graphics produced by GD students	CS	0%	0%	18%	**41%**	**41%**
	GD	0%	0%	**42%**	33%	25%
4. Pleased with implementation of graphics in the app	CS	6%	0%	30%	**41%**	24%
	GD	8%	17%	17%	**42%**	17%
5. GD work significantly improved the visual appeal of the project	CS	0%	0%	12%	35%	**53%**
	GD	8%	0%	8%	**50%**	33%
6. The project overall was significantly better because of GD	CS	0%	6%	12%	**41%**	**41%**
	GD	8%	0%	25%	**58%**	8%
7. Would like to improve the project after the course is over	CS	0%	6%	24%	24%	**47%**
	GD	8%	8%	**33%**	25%	25%
8. Would like to port the project to other platforms like iOS	CS	0%	**41%**	29%	6%	24%
	GD	0%	8%	25%	25%	**42%**
Attitudes about my Teammates						
9. I enjoyed working with my CS or GD team members	CS	0%	6%	12%	**41%**	**41%**
	GD	0%	17%	17%	**33%**	**33%**
10. I would like to work with CS or GD majors again on a future project	CS	0%	0%	12%	24%	**65%**
	GD	8%	25%	**33%**	8%	25%

Table 2: Percentages of CS (*N*=17) and GD (*N*=12) students responding to questions about the project and teammates with largest response category shown in **bold**.

While some GD students obviously felt their work had been ignored or neglected, 82% of CS students and 64% of GD students felt that the overall project was significantly better because of the input from the GD students. A majority of CS (82%) and GD (66%) said they enjoyed working with their teammates from a different major, and only one CS student and two GD students said they did not. When CS students were asked if they would like to work with GD students again in the future, 89% of them indicated they would. GD students were unfortunately less enthusiastic about the prospect; only a third of them said they would like to work with CS students on a future project, a third were neutral about the prospect, and a third said they would not. These results indicate that many GD students found it unrewarding working with CS students and went away from the experience with a bad taste in their mouth. When asked if the students wanted to continue to work on their projects after the semester was over, 71% of CS students said they did, but only half the GD students wanted to. Clearly there were

some root problems that prevented the GD students from enjoying the interdisciplinary work as much as the CS students did.

IDENTIFICATION OF ROOT PROBLEMS

From the survey responses and oral communication with the teams throughout the semester, it was clear that there were several root problems which contributed to friction between CS and GD students in some of the teams:

1. **Communication**. Several CS and GD students indicated that communication was a significant issue. Two students said they would have liked for both classes to meet at the same time to facilitate communication and provide more planned times when students would see each other. Another student said that regularly scheduled team meetings should have been a requirement along with regular teammate evaluations so teammates could be held liable for their work. As noted earlier, some students complained their teammates wouldn't respond to emails or text messages.
2. **Project ownership**. Some CS and GD students lamented the fact that only the CS students chose the project themes, and the GD students had no input. While this mimics the "real world" of graphic design where the designers are assigned the work they must produce, it de-incentivized some GD students because they disliked their assigned project's theme. Some GD students expressed their creative talents were not being fully utilized since they didn't get to participate in coming up with the project themes. It could be that the CS students felt territorial about their project since it was "their" project to begin with and they carried more of the burden to develop a fully functional app. One GD student wrote that they when a disagreement arose about a design issue, the CS teammates tended to do what they wanted.
3. **Valuing others' contributions**. As noted earlier, some CS students did not value the contributions of their GD peers. In some cases the CS students believed they could have produced graphics of the same quality as their GD teammates. When combined with project ownership issues, the GD students could have felt quite marginalized. One GD student complained that her CS teammates did not know "information they should have known" in regards to how apps work on a smartphone. The CS teammates were just learning about mobile apps, and their teammate expected perhaps too much from them.
4. **Deadline mismatch**. Because the GD students received a single project grade, their deadlines were not closely aligned with the CS deliverable deadlines that were scattered across the semester. This made it difficult when CS deadlines approached and the CS team members suddenly required extra work from their GD teammates. For example, a graphic might need to be resized or modified in some way to be implemented correctly. The GD students were busy with other projects, and some of them found it difficult to perform the extra work.
5. **Programmer limitations**. Although the CS instructor tried to evenly disperse programming talent when initially creating the teams, some teams still experienced more technical difficulty than others when implementing their apps. Two CS students noted that they were unable to incorporate the graphics satisfactorily in the time allotted. One team was unable to overcome a key technological barrier during implementation. Overall, six of the eight teams produced fully-functional apps, but two teams had apps that were functionally subpar. As might be expected, the GD students working on these struggling teams recorded some of the more negative responses on the survey.
6. **Android vs. iOS**. The survey revealed a platform "culture gap" between CS and GD students. When students were surveyed if they wanted to port their apps from Android to another operating system like iOS, only 30% of CS students wanted to, but 67% of GD students wanted to make the switch. The written survey comments revealed that CS students preferred Android over iOS because it was free, it used Java, and it supported multiplatform development tools. GD students wrote that iOS

was “better designed” and their preferred platform. This platform-preference divide could have added to the tension between coders and designers.

LESSONS LEARNED

The root problems identified in the previous section make it clear that teaming GD and CS students requires instructors to take a number of steps to ensure a positive experience for all students. First of all, both sets of students need to feel equal ownership of the project. Although the CS students had far more responsibilities in a semester-long project, simply including the GD students in choosing an app theme would have given GD and CS students the feeling that they were both equally responsible for the project’s outcome. It would have motivated some GD students who didn’t like to be assigned a particular project, and it could have even resulted in better app themes.

Improving the lines of communication would also have increased student satisfaction. Although it is difficult to achieve in most universities, ideally the two classes would be scheduled to meet at the same time as was done in [1]. If the courses cannot be scheduled at the same time, both courses should require meeting outside of class several times throughout the semester, perhaps with presentations made by the groups to show their status. Having joint presentations may put added pressure on those CS students who were slow to incorporate GD work into their apps. Synchronizing project deliveries in both courses would also keep both sets of students focused on the same things at the same time. Finally, requiring mandatory meetings, perhaps with an evaluation component, would help keep communication flowing between the teams. It would also prevent procrastination.

Like any long-term project, instructors must keep an active eye on their teams and address any problems early so they do not fester. Instructors may also need to intercede when there is a disagreement about some issue that the teams are unable to resolve. From the outset, students need to know there will be times of disagreement, and they will need to compromise [2].

CONCLUSIONS

Despite the difficulties that a few teams experienced, the combination of CS and GD students into interdisciplinary teams allowed the teams to produce Android apps that far exceeded what the CS students could have done alone. It gave CS and GD students “real world” experience working with students who have different approaches and ideas about app development, and it required them to combine their talents to produce something they were proud to show their peers.

The semester concluded with a well-attended software showcase on a Sunday evening featuring the Android apps. GD students created a display of the app design printed on boards, and the CS students provided several tablets and smartphones running their apps. Awards were presented to two teams that developed apps their classmates judged best in class. Although none of the teams decided to place their apps on the Android Market, some of them were made available for download [8]. The instructors hope to duplicate this interdisciplinary experience again in a future offering of the Android app development course.

REFERENCES

[1] Anewalt, K., Utilizing interdisciplinary teams in teaching e-commerce, *Journal of Computing in Small Colleges*, 19, (2), 288-296, 2003.

[2] Bowen, R., When designers and developers work together, Noupe Design Blog, http://www.noupe.com/how-tos/when-designers-and-developers-work-together.html, July 8, 2010.

[3] Brown, Q., Lee, F., Alejandre, S., Emphasizing soft skills and team development in an educational digital game design course, *Proceedings of the 4th International Conference on Foundations of Digital*

Games (FDG '09), New York, NY: ACM, 240-247, 2009.

[4] Carter, L., Ideas for adding soft skills education to service learning and capstone courses for computer science students, *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education (SIGCSE '11)*, New York, NY: ACM, 517-522, 2011.

[5] Catanio, J. T., An interdisciplinary practical approach to teaching the software development life-cycle, *Proceedings of the 7th Conference on Information Technology Education (SIGITE '06)*, 3-8, 2006.

[6] Ebert, D. S., Bailey, D., A collaborative and interdisciplinary computer animation course. SIGGRAPH Computer Graphics 34, (3), 22-26, August 2000.

[7] Goadrich, M. H., Rogers, M. P., Smart smartphone development: iOS versus Android, *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education (SIGCSE '11)*, 607-612, 2011.

[8] McCown, F., "Android app course wrap-up", http://frankmccown.blogspot.com/2011/12/android-app-course-wrap-up.html, December 20, 2011.

[9] McCown, F., Android app development tutorials on Google Developers University Consortium, https://developers.google.com/university/courses/mobile#mobile-computing, 2012.

[10] Reed, J. Janzen, D. S., Contextual Android education, *24th IEEE-CS Proceedings of Software Engineering Education and Training (CSEE&T)*, 487-491, 2011.