

# FACILITATING A COURSE IN TABLET PC SOFTWARE DEVELOPMENT

**Roy P. Pargas**

*Department of Computer Science, Clemson University*

*pargas@cs.clemson.edu*

*<http://www.cs.clemson.edu/~pargas/tabletpc>*

## 1 ABSTRACT

This paper describes a Tablet PC software development course first taught in the fall 2005 semester. The approach taken was different from typical software development courses but it succeeded in enabling the students to develop large Tablet PC applications and providing them with opportunities to present their work in formal settings.

## 2 PROBLEM STATEMENT

This course was going to be an *adventure*, although we didn't know it when we started. Our objective was simple enough: *we wanted to learn how to write Tablet PC software*. Our strategy was uncomplicated: *we would learn from examples set forth in our textbook and from a recently completed computer science masters project*. We wanted our software to live and be useful beyond the end of the semester, and so we decided that *each Tablet PC project must serve to support the teaching or learning of a specific academic topic*. It could be a learning tool for students trying to master a topic or a teaching tool for instructors trying to deliver course material, or both. Beyond that, we had no specific plans or expectations. We played it by ear from week to week and, as we happily report below, things turned out very well.

## 3 PROPOSED SOLUTION

### 3.1 Goals and General Approach

We started the semester with several goals: (1) to get our Tablet PC programming feet wet by attempting several small weekly programming exercises, (2) to propose, design, and implement a large Tablet PC application, (3) to document the application with a User's Manual and a Technical Reference Manual, and (4) to gain experience with formal presentations through a class mini-conference at the end of the semester with other students and faculty as guests. Extremely good fortune enabled us to add two goals mid-semester: (5) to submit for each application a proposal for a poster presentation at WIPTE 2006 and, if accepted, (6) to present the poster at the conference.

From the outset, the instructor announced that he would not try to *teach* the students how to write Tablet PC software. Rather he and the students would *learn* together. Indeed, his role was more of a *facilitator* than an instructor, hence our choice for the title of this paper. It was fortunate that Jarrett and Su [1] had written an excellent textbook for such an approach. The text is easy-to-read, comprehensive, practical, humorous, and most

important, chock-full of sample code. It is excellent both in a class and for an individual who wants to learn on his or her own.

The instructor and students (initially nine, later eight) met for at most 2-1/2 hours, every Tuesday evening. Students had laptop computers on which they could develop Tablet PC code. Two of the student computers were Tablet PCs and two additional Tablet PCs were available for the other students to borrow. The class consisted of six computer science and three computer engineering students. These were advanced students; all had taken computer science courses well past data structures (CS4). They were also very good students; two were double-majors, one had a minor in Russian, and three were in the undergraduate Honors Program. Seven of the students were undergraduates and two were graduate students.

The instructor had in mind the following grading scheme for the course: small programming assignments 30%, project proposal 10%, project software 50%, documentation 20%, and final presentation 10%. In reality, the grading scheme was not used because none of the students needed the threat of a poor grade to encourage them to work on their assignments. The course was not required and the students took the course not because they had to, because they wanted to.

The students were told that, unlike previous courses they had taken, in this course they were allowed, indeed *encouraged*, to share with each other what they learned about what works and what doesn't. This included sharing code. Furthermore, each student was to post all assignments on a webpage, including all source and documentation. Thus, whenever a student demonstrated software, other students were able to download and run the software for themselves.

### 3.2 Weekly Schedule

Table 1 summarizes the course activities for the semester. In the first class meeting, students installed the Microsoft Tablet PC SDK on their laptops. They also installed software files required to run *DSInk* [2] which served as a model for the projects that the students would eventually write. Class activities during the first six weeks were spent on Jarrett and Su, alternating between studying (in odd-numbered weeks) the examples provided in the text, and students demonstrating through small programs (in even-numbered weeks) that they understood the material enough to use them in working code.

For each of the first three assignments, the students were instructed simply to implement three of the programming features demonstrated in the text for that week in an application of their own design. Injecting humor in their programs earned kudos from their peers. The results were varied and entertaining. Students demonstrated their programs, indicating which programming features they selected and explaining the code that provided the features. Students were asked to post their weekly assignments on the web making all of their code available to other students.

Starting from week 6 through the end of the semester, all activities focused on the semester project. Students had been informed from the beginning that they were to consult with the instructor privately about what they wanted to do. The assignment for week 7 was to write a one-page proposal for a semester project. However, when the

students learned of the decision by the WIPTE 2006 organizing committee to accept poster presentations, they all agreed to develop proposals for poster presentations as well.

This redefined the course activities for weeks 7 through 10. Each week, students submitted one-page project draft proposals for a WIPTE 2006 poster presentation written in accordance with conference specifications. In addition, students submitted the initial software framework for their semester project and demonstrated incremental progress. The instructor critiqued and edited the proposals, and returned them to the students (on paper when class met and through the web using Microsoft Journal during fall break when class did not). After three weeks and five revisions, the one-page proposals for poster presentations were ready and were submitted to WIPTE 2006.

<b>Week</b>	<b>Read</b>	<b>In-Class Activity</b>	<b>Homework: Exercises / Project</b>
1	1-4	Software setup. Discuss Ch. 1-4. Study J&S Ch. 4 examples.	Implement 3 features from Ch. 4
2	5	Students demo Ch 4 assignment	
3		Study J&S Ch 5 examples	Implement 3 features from Ch. 5
4	6, 7	Students demo Ch 5 assignment	
5		Study J&S Ch 6 and 7 examples	Implement 3 features from Ch. 6, 7
6		Students demo Ch 6,7 assignment	<i>Project proposal, WIPTE proposal</i>
7-9		<i>Demo/critique software</i> <i>Discuss/critique proposals</i>	<i>Software design/implementation</i> <i>Revise WIPTE poster proposals</i>
10		<i>Demo/critique software</i>	<i>Submit WIPTE poster proposals</i>
11-13		<i>Demo/critique software</i>	<i>Advance/improve project software</i>
14		<i>Practice presentation</i>	<i>Develop Final WIPTE poster</i>
15		<i>Final poster, presentation</i>	<i>Submit all final documentation</i>

**Table 1.** Weekly activities in this course. *Project* activities are in *italics*.

This activity was made a bit more complicated by the fact that the proposals for posters had to be written in the past tense, as if the projects were already completed. At the same time, the design and implementation of the projects were just getting underway. Both the instructor and the student had to balance the feasibility of the projected software as evidenced by the initial implementation of the code, with the claims being made in the proposal. Great care was taken not to promise in the proposal more than what the student was confident in delivering. Simultaneously, equal care was taken not to shortchange the project, i.e., not to offer less in the proposal than the student was capable of producing.

Weeks 11 through 13 were spent reviewing individual projects. During each meeting, students demonstrated the current versions of their software. The instructor met with each student for 30 minutes discussing approaches, noting problems, suggesting possible solutions, and charting progress. The primary objective during these weeks was to insure that the students were making sufficient progress to meet end-of-semester deadlines. It was especially during this time that the instructor served as facilitator, mentor, critic, and guide. Other students were welcome, but were not required, to attend these advising sessions. If something of potential value to entire class was discussed, the

information was emailed by the instructor to all students. As always, intermediate versions of student projects were posted on the students' websites, available for all to see.

Practice presentation day came in week 14. Students gave 25-minute formal presentations of their projects. They were asked to practice presenting by themselves a minimum of five times prior to this meeting. The instructor provided feedback on presentation style and content, providing tips on everything from voice projection, eye contact, speed of delivery, body English, speech mannerisms, and basic grooming. This was to prepare for final presentation day in week 15.

On presentation day, students were asked to wear suits and ties and Sunday dress. Invitations had been issued the week before to faculty and other students. A pizza-and-soft-drink reception immediately after the presentations was promised with the aim of boosting attendance. Each student was allowed 25 minutes of presentation time, including 10 minutes of PowerPoint slides, 10 minutes of software demonstration, three minutes of questions, and two minutes of setup/breakdown. The instructor, serving as session chair, introduced the speakers, monitored the time, took live pictures and kept the mini-conference flowing smoothly. The students also had posters of their work displayed around the classroom for visitors to view before and after the mini-conference. During this final week, the best projects were considered for further development and possible *paper* submission to future using-technology-in-education conferences.

### 3.3 Summary

In retrospect, the course went through four overlapping phases: (1) development of programming skills through emulation and experimentation in weeks 1-6, (2) project proposal and preliminary design in weeks 7-10, (3) project development in weeks 9-15, and (4) final presentation and project submission in weeks 14-15. There were two unique highlights in the course: submission of poster presentations to WIPTE 2006 and the activities leading to the final presentations at the end of the semester. These two activities provided students with real-life academic research experiences not available in other courses they had taken.

## 4 WORK IN PROGRESS: EVALUATION PLANS

This paper reports on a Tablet PC software development course, not on the use of a Tablet PC in teaching or learning course material. Hence, we cannot report on such metrics as "improved learning outcomes" or "student attitude changes". Instead, we asked students to rate 10 statements using a Likert scale: (a) Agree very strongly, (b) Agree strongly, (c) Agree, (d) Disagree, (e) Disagree strongly, (f) Disagree very strongly.

The first five statements all ended with "... helped me learn how to write Tablet PC software": (1) Reading the text by Jarrett and Su ..., (2) Developing weekly programming assignments based on a chapter of the book ..., (3) Presenting my weekly assignments to the rest of the class ..., (4) Listening to other students' presentations of their weekly assignments ....., and (5) Discussions with other students about their assignments .... The next three statements involved the poster proposal exercise: (6) Developing a poster proposal for WIPTE 2006 helped me design my semester project, (7) The possibility of presenting at WIPTE 2006 motivated me to work harder on my semester project, (8) Developing a poster proposal for WIPTE 2006 distracted me from

working on my semester project. Finally, the last two statements had to do with posting evaluations on the web: (9) Seeing the evaluation of other students' assignments helped me improve my own work, (10) Display on the web of the evaluation of my work embarrassed me.

<i>N</i>	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
(a)	3	3	1	1		3	5		2	
(b)	3	3	2	2	4	3	1		3	
(c)	2	1	4	5	3	1	1	1	1	
(d)		1				1	1	2	1	1
(e)								2	1	1
(f)								3		6

**Table 2.** Mid-semester assessment results

Although the number of respondents is small, the results of the survey taken in mid-semester and shown in Table 2, are very encouraging. Responses to the overall format of the class (questions 1-5) are almost all positive and at times very strongly positive.

The WIPTE conference and the possibility of presenting their results served as a motivator for working harder on their projects (questions 6-8). Only one student thought that the WIPTE proposal did not help in design the large project and was a distraction from their project. Public evaluation of other students' work was viewed as mostly helpful (two disagreed). And public evaluation of one's own work was not at all viewed as an embarrassment, a persistent concern of the instructor until this survey. We plan to ask the questions again at the end of the semester and to combine and compare results.

## 5 FUTURE WORK

As the result of our experiences, we hope to be able to teach this course on a regular basis every fall semester. Fortuitous events such as the opportunity to write poster proposals and possibly the present posters at WIPTE significantly enhanced the student learning experience. If WIPTE issues a call for papers and posters at about the same time each year, i.e., early November, then responding to both calls will become a regular (and prominently advertised) feature of this course.

## 6 ACKNOWLEDGMENTS

This work has been generously supported by a 2005-2006 grant from the Microsoft Tablet PC and Computing Initiative (Jane Prey, Program Manager) and by a Tablet PC donation by Hewlett-Packard Company (Rob Reed, University Relations and Wayne Johnson, VP University Relations). We also wish to thank the students of this course Danielle Deutsch, Adam Goodbar, Amit Jain, Himanshu Kumar, Sarah Peck, Dhaval Shah, Achal Singhal, and Jerry Stasulis for their patience, energy, enthusiasm, and good-natured humor: all essential qualities when embarking on an adventure.

## REFERENCES

1. Jarrett, R. and Su, P., *Building Tablet PC Applications*, Microsoft Press, Redmond, WA, 2003
2. Culler, R. *DSInk: A Software Library for Developing Data Structures Applications for the Tablet PC*, M.S. Research Paper, Department of Computer Science, Clemson University, Clemson, SC, 2005