TABLET PCs IN A CODE REVIEW CLASS

Technology in Practice strand

Sam Kamin
Computer Science Dept.
University of Illinois
kamin@illinois.edu

Wade Fagen
Computer Science Dept.
University of Illinois
wfagen2@illinois.edu

Abstract

We describe an intervention in a code-review class using Tablet PCs. The class, Programming Studio, is a required class for Computer Science students at the junior level, mainly consisting of weekly small-goup meetings — 3–6 students, plus a moderator — in which each student presents his or her program for discussion. In the traditional format, each student presented their program by displaying it on the class display. In the Tablet PC format, every student has a copy of the presenter's program on their tablet; students can draw on the program, with all students seeing every student's annotations, and can navigate independently in the program. We discuss how the tablets are used, and present the results of an end-of-semester survey indicating that students who participated using both studio formats found the tablets helpful.

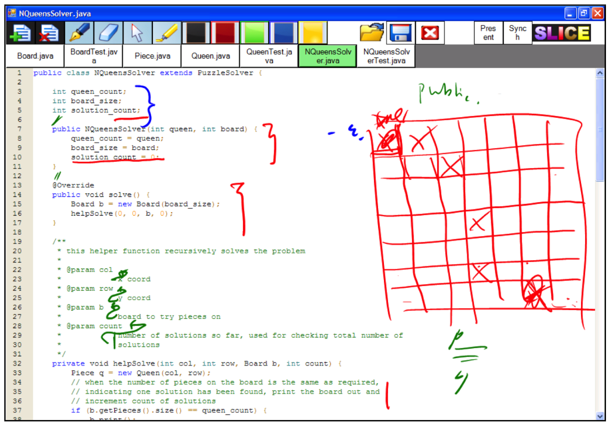
1 Problem Statement and Context

Numerous authors have reported on uses of pen-enabled computers in classrooms, both large and small [1,2,6]. In this paper, we report on the use of Tablet PCs in a somewhat different educational environment: a code review class. Tablet PCs can enhance code reviews by making it easier to point to locations in the code, draw pictures to represent data structures and algorithms, and sketch out alternative coding methods.

Students in the Computer Science department at the University of Illinois (Urbana-Champaign) take CS 242, Programming Studio, typically in the first semester of the junior year. Its purpose is to to build individual programming skills by having students do a lot of programming and, more importantly, present their program publicly for discussion each week. Discussion sections have 3 to 6 students, plus a moderator; each presenter plugs into the room display and discusses their program for fifteen to twenty minutes.

In describing the Programming Studio in 2007 [7], we noted, "We need to continually work on maintaining the quality of the discussion in the discussion sections." Programs are complicated, and it is hard to have a stimulating discussion on something that only one of the interlocutors understands. In our observations, we often found that only one or two

(a) A studio section using Tablet PCs



```
public class NQueensSolver extends PuzzleSolver {

    int queen_count;
    int board_size;
    int solution_count;

    public NQueensSolver(int queen, int board) {
        queen_count = queen;
        board_size = board;
        solution_count = 0;
    }

    @Override
    public void solve() {
        Board b = new Board(board_size);
        helpSolve(0, 0, b, 0);
    }

    /**
     * this helper function recursively solves the problem
     *
     * @param col
     *            x coord
     * @param row
     *            y coord
     * @param b
     *            board to try pieces on
     * @param count
     *            number of solutions so far, used for checking total number of
     *            solutions
     */
    private void helpSolve(int col, int row, Board b, int count) {
        Piece q = new Queen(col, row);
        // when the number of pieces on the board is the same as required,
        // indicating one solution has been found, print the board out and
        // increment count of solutions
        if (b.getPieces().size() == queen_count) {
```

(b) Tablet PC screenshot

Figure 1: Images from a CS 242 section

reviewers could completely follow the presentation, while the others got lost at some point and then lost interest. The Tablet PCs were introduced as a way to help the discussions be more engaging and efficient.

Without tablets, the studio set-up is this: The students and moderator sit around a table, with a large monitor at the front, and each student takes a turn presenting their program. With tablets, the physical arrangement is the same, but each student has a Tablet PC (see Figure 1(a)). Each presenter loads their program (usually in multiple files), and it shows up simultaneously on all the tablets. As the presenter explains the program, everyone can write on it, with all the writing being shared. In other words, the students have a shared whiteboard with the current presenter's program acting as the background image. Furthermore, since each tablet has a separate copy of all the code, each student can explore it independently of the presenter; if the presenter goes too quickly over a portion of the code, a reviewer can remain on that portion while the presenter moves on.

2 Method Employed

The system we introduced in the Studio is a straightforward "shared whiteboard" application, developed using the SLICE framework [3,5]. As each student takes a turn as presenter, he or she loads their program files into the system, where each is represented by a tab. Within each program, the text of the file forms the background of the display; the text cannot be directly edited, but can be annotated upon. All ink strokes entered by any student are immediately shared by all. Figure 1(b) shows the SLICE screen during a studio discussion.

The Tablets provide two capabilities: (1) The ability to write, draw, and point. Drawing is useful to explain the deep structure of a program, data structure, or algorithm; writing makes it easier for reviewers to ask "what if you did it this way?" questions; pointing makes it easier to draw attention to a particular spot in the program. (2) The ability to explore the presented program independently of the presenter. Each student can switch tabs and scroll through the files in the presented programs. This allows a reviewer to spend time understanding a section of code after the presenter has moved on, or simply to explore it in a different order than the presenter uses. (A button on the interface allows a student to synch up with the presenter.) This can allow a reviewer to gain a better understanding of the presented program — or at least of some part of it — and thus make more substantive comments about it.

The screenshot in Figure 1(b) comes from a Thursday evening section with six students (plus a moderator). The screenshot is taken at the end of the discussion. (Lack of space prevents us from illustrating the sequencing of the annotations.) The assignment was the well-known eight queens problem. This student's program consisted of seven files (represented by the tabs just below the top toolbar), and we are looking at `NQueensSolver.java` (the green tab). We can see that the presenter drew a chessboard while explaining his program. This image contains ink marks from four of the participants. Timestamps on the ink strokes indicate that this part of the presentation took about five minutes.

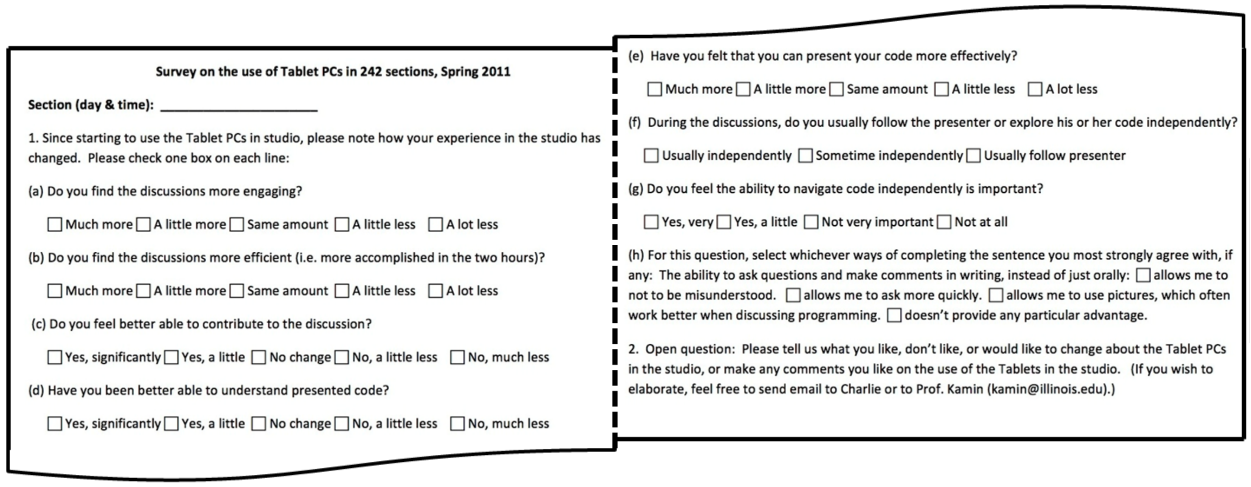
This screen shows several features typical of the use of the tablets:

- Drawings are often used to explain programs.
- On the other hand, the majority of pen strokes are used simply to point (underline, circle, etc.). When using tablets, students would refer orally simply to "this line" — indicating the actual line with the pen — whereas without the tablet various locutions were used to identify places in the program. This is one way the tablets made the discussion more efficient.
- The interface allows the students to choose from a (small) palette of colors, but does not assign different colors to them. In practice, the students tended to quickly differentiate themselves by colors. Of the four participants who made annotations on this file, the presenter used red, the moderator green, one other student green, and the fourth used two colors (red and blue).

This figure does not show any examples of participants writing code, which was also common. It also does not show any uses of "laser strokes." The laser stroke is a feature of the interface allowing someone to point to something with a heavy red line which then disappears when anything else is drawn; it is a way to point to items without leaving the screen cluttered.

One observation (for which we have no quantified data) is that students very quickly got into the habit of looking at the code on their own Tablets rather than on the room display. This may be because the tablets were simply easier to read, or because the students were taking advantage of the opportunity to review the presenter's code "out of order." Furthermore, the presenters *always* wrote on the tablets as they presented.

To offer just a few summary statistics on this meeting: During the 70 minutes of the meeting, the participants annotated programs with a total of 852 ink strokes. These strokes



Survey on the use of Tablet PCs in 242 sections, Spring 2011

Section (day & time): ____________________

1. Since starting to use the Tablet PCs in studio, please note how your experience in the studio has changed. Please check one box on each line:

(a) Do you find the discussions more engaging?

☐ Much more ☐ A little more ☐ Same amount ☐ A little less ☐ A lot less

(b) Do you find the discussions more efficient (i.e. more accomplished in the two hours)?

☐ Much more ☐ A little more ☐ Same amount ☐ A little less ☐ A lot less

(c) Do you feel better able to contribute to the discussion?

☐ Yes, significantly ☐ Yes, a little ☐ No change ☐ No, a little less ☐ No, much less

(d) Have you been better able to understand presented code?

☐ Yes, significantly ☐ Yes, a little ☐ No change ☐ No, a little less ☐ No, much less

(e) Have you felt that you can present your code more effectively?

☐ Much more ☐ A little more ☐ Same amount ☐ A little less ☐ A lot less

(f) During the discussions, do you usually follow the presenter or explore his or her code independently?

☐ Usually independently ☐ Sometime independently ☐ Usually follow presenter

(g) Do you feel the ability to navigate code independently is important?

☐ Yes, very ☐ Yes, a little ☐ Not very important ☐ Not at all

(h) For this question, select whichever ways of completing the sentence you most strongly agree with, if any: The ability to ask questions and make comments in writing, instead of just orally: ☐ allows me to not to be misunderstood. ☐ allows me to ask more quickly. ☐ allows me to use pictures, which often work better when discussing programming. ☐ doesn't provide any particular advantage.

2. Open question: Please tell us what you like, don't like, or would like to change about the Tablet PCs in the studio, or make any comments you like on the use of the Tablets in the studio. (If you wish to elaborate, feel free to send email to Charlie or to Prof. Kamin (kamin@illinois.edu).)

Figure 2: Survey instrument

were not added in uniform numbers by all participants. The most active annotator was the moderator (351 strokes); the most active student added 212 strokes and the least active only 10. Most students wrote most often during their own presentations (though there were exceptions), but varied greatly in the amount of writing they did during other students' presentations; the most active student added 102 strokes during other students' presentations, and the least active just 7.

3 Results and Evaluation

To assess the Tablet PC intervention, we did a study in six sections of CS 242 (23 students and 4 moderators). We had students follow the traditional structure for six weeks, then use the tablets for four weeks, and then switch back for three weeks. At the end of the semester (that is, in week 14), we surveyed the twenty-three students in those sections.

The survey instrument was a single page, with nine questions. The first seven were on a sliding scale, the eighth allowed multiple selections from a list, and the last was an open-ended written response question. Figure 2 shows the survey instrument. Figure 3 shows the results for all questions except the last; we summarize the results on the latter below.

In retrospect, on some questions we should have offered more responses; for example, on questions a, b, and e, we should have had an answer in between "somewhat more" and "much more." Nonetheless, what is striking about the responses is their near unanimity:

- Questions a through e ask, in various forms, whether the students found the tablets helpful or harmful. On questions a and c, not one of the twenty-three students found them harmful, and on d and e, only one student found them harmful. With the exception of b, a signifcant majority found them useful. On question c, all but two students found that the tablets made it easier to contribute to the discussion.

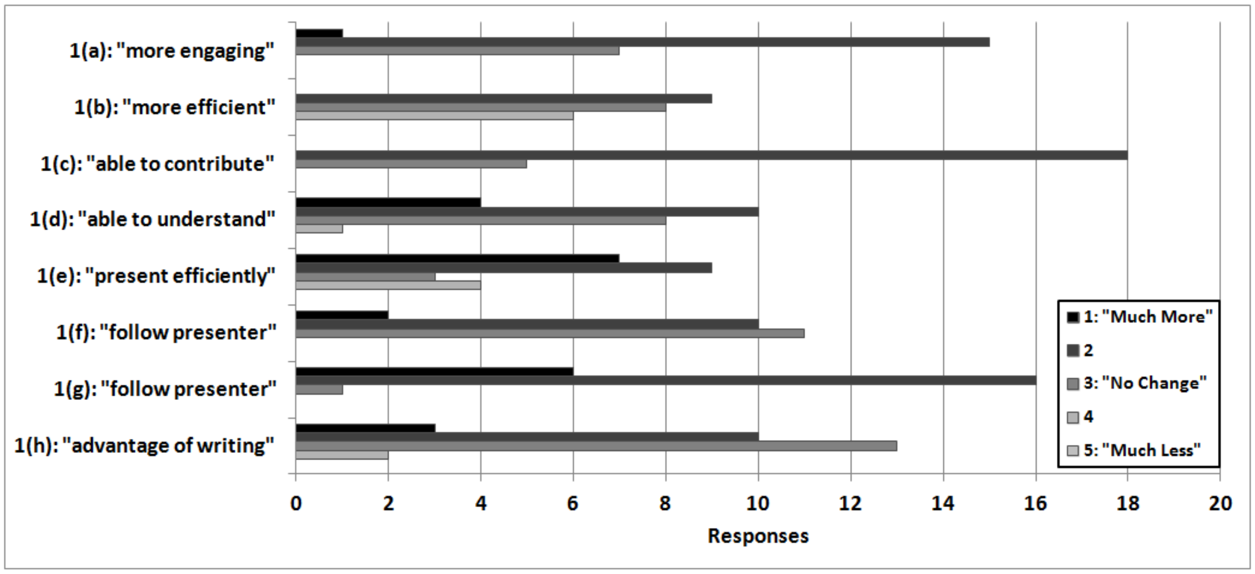


Figure 3: Survey results. (The inset gives response choices for a–e.)

- Responses to question b show that students were split on the question of whether the discussions using the tablets were more "efficient." We surmise that technical issues — additional set-up time, and occasional disruptions caused by bugs in the software — detracted from the efficiency of the meetings.
- On question h, concerning the advantages of writing questions versus speaking them, not one student chose "no advantage." (Note that this question allowed for multiple selections, so the total number of responses is more than twenty-three.)

Question 2 was an open-ended text question requesting feedback on the system. Most of the comments related to shortcomings in the system itself, three in particular: lack of syntax highlighting (mentioned by six students; since added); not being able to present within their normal IDE, or be able to run code (five students); and bugs (two students). There were several positive comments (mainly redundant with questions g and h), and one specific suggestion for a new feature other than syntax highighting. The answers to this question do demonstrate that students were perfectly willing to make critical comments about the system, suggesting that their responses in question 1 were honest.

4 Future Work

We have presented an application of Tablet PCs in a code review class. The problem we confronted with this class is that reading other people's code is taxing and may be tedious. The Tablet PCs were introduced as a way to ameliorate this problem. In this paper, we have explained the studio set-up both with and without the tablets, discussed how the students used the tablets, and presented the results of a survey showing that students generally preferred using the tablets over the traditional organization of the studio.

During the study, we also took audio recordings of the studio. With each student wearing a microphone and recording into a separate audio stream, we have a total of over 195 hours of recordings. In [4], we present a mechanical analysis of these recordings, defining a measure we call "active engagement" (related to "turn-taking"). We show that there is a statistically significant increase on this measure between tablet and non-tablet meetings of the studio, which is consistent with, and tends to confirm, the findings of the survey reported here.

5 Additional Resources

SLICE [3,5] is a framework for developing tablet applications for education. We have used it to build several in-class applications as well as the code review application described here. Furthermore, it is substantially device-independent, meaning that a single codebase (which is to say, a single set of Javascript functions) can run the same application on Microsoft Tablet PCs and on Android tablets. The SLICE system is freely available at **slice.cs.illinois.edu**.

6 Acknowledgements

We thank Chris Cortez for helping us run this experiment. Studio instructors Mike Woodley and Charlie Meyer were instrumental in getting us access to the class and gaining the cooperation of the studio moderators.

7 References

[1] Anderson, R. J., Anderson R., Simon, B., Wolfman, S. A., VanDeGrift T., Yashuhara, K. Experiences with a Tablet PC Based Lecture Presentation System in Computer Science Courses. Proc. 35th SIGCSE. Norfolk, Va. 2004. 5660.

[2] Berque, D. Pushing Forty (courses per semester): Pencomputing and DyKnow tools at DePauw University. in WIPTE '06. Purdue University Press. 2006.

[3] W. Fagen, S. Kamin, A Cross-Platform Framework for Educational Application Development on Phones, Tablets, and Tablet PCs, Proc. 2012 Intl. Conf. on Frontiers in Education: Computer Science and Computer Engineering (FECS'12), July 2012, Las Vegas, NV.

[4] W. Fagen, S. Kamin, Measuring Increased Engagement Using Tablet PCs in a Code Review Class. *SIGCSE*, Denver, CO, March, 2013.

[5] S. Kamin, M. Hines, C. Peiper, B. Capitanu, A System for Developing Tablet PC Applications for Education. *SIGCSE*, Portland, OR, March, 2008.

[6] S. Kamin, W. Fagen, Supporting Active Learning in Large Classrooms Using Pen-enabled Computers. Proc. 2012 Intl. Conf. on Frontiers in Education: Computer Science and Computer Engineering (FECS'12), July 2012, Las Vegas, NV.

[7] M. Woodley, S. Kamin, Programming Studio: A Course for Improving Programming Skills in Undergraduates. *SIGCSE*, Covington, KY, March, 2007.