Pairs Programming
Scott D. Anderson

Introduction
Over the past few years, I’ve heard a lot about “pairs programming” and the related concept of “extreme programming.” The basic idea is that two people work together as a team during coding. However, it’s different from what you might think. Most group work, which is standard in any professional programming (research, industry and so forth), the coding tasks are divided up and teach team member works solo on his or her own tasks, such that the aggregate effort gets the job done. In pair programming, on the other hand, the two people in the pair work together at the same computer at the same time. One takes the keyboard and mouse (the driver) while the other comments (the navigator) and then they switch. They aren’t dividing up the tasks: they have two people working on the same task. If standard group work is a baseball team, a collection of individual efforts, pairs programming is more like dancing (ballroom or swing dancing, not tap dancing). You can learn more about it from some of the following web sites:

http://pairprogramming.com/
http://collaboration.csc.ncsu.edu/laurie/
http://c2.com/cgi/wiki?PairProgramming

Advantages
Industry wouldn’t even consider paying two people to do the work of one unless it conferred great benefits, and it does. In a number of studies, pairs programming produced better code in less time. In fact, the productivity of a pair is often more than twice what someone working along would achieve. So, for example, a pair might finish a programming task in 40 hours that one person working alone would take 80 hours to do, and the pair’s code would have fewer bugs. Therefore, it becomes cost effective to pay two people to work as a pair.

Some college and universities are introducing pairs programming in their classes, and they are finding that each student in the pair learns as much as if they worked solo, and programming assignments take less time and yield better quality. An additional advantage is that many students find it to be more fun and rewarding.

The jury is still out, though, and many people are not yet convinced of the improvements in productivity and quality. Indeed, I’m not yet convinced. Nevertheless, I’m very intrigued, and I’d like to consider it for this class.

Proposal
I propose to allow, but not require, pairs programming in this class. Please understand that I’d like a real commitment to the technique. I don’t want people dividing the work or having one person work on Tuesday night and the other work on Wednesday night. When the program is being worked on, both people have to be working on it. That means that the single biggest problem we will face is busy and incompatible schedules.

It may be that you’d like to try pairs programming, but that there’s no one in the class that can pair with you, because of your busy schedule. It might be that an odd number of people want
to try it, and you’re the odd person out. If something like that happens, you’ll have to work solo. It may be that you’d rather not try pairs programming; that’s certainly understandable.

Method

I suggest that we re-pair each week. This allows people to try pairs programming for one assignment but not for the semester. It may be that people like it and more and more people will want to try it. It may be that some will try it and decide they don’t like it. By re-pairing every week, we can allow for changes in interest.

I also think we should try different partnerships each week. That means that if one pair doesn’t work out because, despite their best efforts, their schedules just weren’t compatible, each person can still try pairs programming the next week with a different person. Or, perhaps there’s just a personality conflict.

Of course, re-pairing each week means a lot of overhead for coordination. You might think about writing your schedule out on a piece of paper, so you can make copies and give it to prospective partners. Or, perhaps people can post their schedules to the class conference. I’m open to your ideas on how we can make this work.

Another aspect of the commitment I’m seeking is to the way that pairs programming works. Neither person is passively watching or resting: both are actively programming the whole time, bouncing ideas off each other and suggesting solutions. If more than 15 seconds of silence go by, the navigator needs to get more involved. (The driver is always involved, since she has the keyboard and mouse.) Switch drivers often, though I understand that if one person is a much more capable keyboardist, you may prefer to have her drive.

Finally, of course, I’m looking to a commitment to learning to program as the main goal, not the completion of a programming assignment. This is really no different from being in industry, where you should always be seeking to increase your own skills.

Conclusion

I think pairs programming is a promising and interesting idea, but I will not be disappointed (okay, maybe a little) if there is no interest. I won’t be upset if the experiment dies out during the semester. So, please be honest about what you’d like and how we can best make this work, if we decide to try.