Code Warriors and Code-a-Phobes: A Study in Attitude and Pair Programming

Lynda Thomas, Mark Ratcliffe and Ann Robertson
Computer Science Department
University of Wales, Aberystwyth
Great Britain SY23 3DB
{ltt,mbr,iar}@aber.ac.uk

Abstract

This paper reports on how first-year students who have programmed before see their programming interest and ability and how this self-perception relates to their performance in the introductory programming course. In particular we examine how this self-perception is reflected in their reactions to the pair-programming technique for developing software.

Students who had programming experience before University were given a survey that placed them on a scale that we have called Code Warrior to Code-a-phobe. We then placed them in 'opposite' and 'similar' pairs for a pair programming exercise and surveyed their reactions. There was evidence that students who have considerable self-confidence do not enjoy the experience of pair programming as much as other students and that students produce their best work when placed in pairs with students of similar self-confidence levels.

Categories and Subject Descriptors

K.3 [**Computers & Education**]: Computer & Information Science Education – *Computer Science Education.*
D.2 [**Software Engineering**]: Coding Tools and Techniques – *Object-oriented Programming*

General Terms

Human Factors

Keywords

Pair programming, Self-confidence, First Year Programming, CS1, Closed Labs.

1 Introduction

In our search at Aberystwyth for ways to make our courses more interesting and appealing to the wider range of students who now enter higher education, we are in the process of developingan educational environment that allows students to maintain control of their own learning by providing a suite of resources that bundles traditional pedagogical approaches such as lectures, tutorials and seminars, with various other innovations [5]. Our consideration of how to improve the first year sequence and other courses has also led us to consider the issue of individual learning styles [6] and their impact on teaching and student success. This in turn has led to an interest in how students' individual characteristics impact on the somewhat standardised way that we expect them to work.

In this paper we examine correlations between a specific characteristic - attitude to programming - and performance on an introductory programming course. In particular, we examine whether attitude to programming impacts on the use of pair-programming in closed labs.

2 Pair Programming

Pair programming is "a programming technique where two people program with one keyboard, one mouse and one monitor" [1]. It has gained a good deal of interest recently, probably because of its inclusion in the set of techniques that comprise Extreme Programming. Initially the evidence for the effectiveness of pair programming was largely anecdotal, but recently surveys in industrial [9] and educational [4,10] settings seem to indicate that most respondents find it enjoyable and that it improves the level of code quality.

Williams and Kessler's reports on reaction to pair programming in their courses are certainly encouraging:

- 95% of the students agree with the statement "I was more confident in our assignments because we pair programmed." This was borne out in their programs, which passed 15% more of the test cases
- 84% of the students report enjoying the experience of programming more when working in a pair [10].

As educators, we wanted to see if these results could be duplicated with our own programming students. The results certainly fit in with the collaborative view of programming practice that we have held dear since reading *The Psychology of Programming* [8] many years ago, and we could certainly believe that this practice might work well with mature programmers in industry, but we had niggling doubts as to how well it would work with 18 year olds straight from high school.[1]

SIGCSE'03, February 19-23, 2003, Reno, Nevada, USA.

[1] The origin of these doubts was Dr. Williams' description of the average student at the University of Utah. Many of these students have postponed University while they completed a missionary year. In addition, many are married. This is quite different from the profile of

3 Code Warriors

The use of the term 'code-warrior' in this work comes from a paper by Debora Weber-Wulff that was presented at ITiCSE 2000 [7]. She describes certain students as seeing themselves "…as a sort of code warrior, fighting with the enemy compiler, forcing it to assent to their glorious code and to produce a program that obeys their every desire." In this paper Weber-Wulff concentrates on how these students (who often are not actually as advanced as they think) can be led into understanding that production-quality large software systems require documentation, deep understanding, and team programming. The term certainly describes many of our students at Aberystwyth, particularly those in their first and second year.

The term 'code-a-phobe' we have coined ourselves after observing what seems to be an unfortunately common phenomenon among our students. Many students decide to major in computer science but seem to obtain little satisfaction from programming. The first author first noticed this about 10 years ago whilst teaching in the States; but this phenomenon now seems to be equally common among British students. An unkind observer might wonder what these students were doing in computer science programs at all, but we are forced to admit that many jobs in the IT industry do not require highly developed programming skills. These students often aspire to these sorts of jobs – or they simply enjoy manipulating software and hardware and a computer science degree seems like a good idea to them. They frequently report 'hating to program' or 'being hopeless at programming'. Whether we like it or not, as educators we are faced with these kinds of students and the combination of them in classes with 'code-warriors' makes programming courses very difficult to teach [2].

The phenomenon of the two sorts of students mentioned above is not just an attitudinal one. A recent multi-national study on the assessment of programming skills of first year CS students that was coordinated at ITiCSE 2001 investigated the programming competency of students from four universities in three different countries [3]. Results were sobering. The average score on an assigned programming problem was 22.89 out of a possible 110 points. When the data was analysed in more detail it appeared that performance was bi-modal. Many students never got going with the problems at all, others did very well.

4 A Caveat

We realise that there are actually two variables here: attitude and performance. We have no way of knowing if the 'code-warriors' described by Weber-Wulff are the sorts of students who actually *performed* well in the programming skills study (see also section 6.2). The students whom we have labelled 'code-a-phobes' might actually be perfectly competent programmers. We look at this in section 6 but gain no statistically conclusive evidence. We need to perform more research to determine the relationship between attitude and performance. We consider that the experiment described below is a step in this direction and are currently continuing to pursue this research.

5 The Experiment

In this study we attempted to examine the practice of pair programming in the context of the distribution discussed above. At the beginning of the semester we asked first-year students who had prior programming experience to complete a 'programming attitude' questionnaire. This questionnaire was adapted from one originated by Davis and others at the University of Southampton, where it was successfully used to place students in appropriate groups for first year instruction [2]. On this instrument students place themselves on a scale from 1 to 9, where we described:

9 as:

- **Code-Warrior** I have had no trouble at all completing programming tasks to date, in fact they weren't challenging enough. I love to program and anticipate no difficulty with this course.

and 1 as:

- **Code-a-Phobe** I don't like programming and I don't think I am any good at it. I can write simple programs but have trouble writing new programs for solving new problems.

Later in the semester, the students were given a lecture on pair programming and a short paper to read to prepare themselves for this exercise. They were then placed in pairs to solve a simple problem during a regularly scheduled practical laboratory. The pairs were not assigned randomly (see below). The students were then surveyed for a reaction to this experience and the programs they produced were read and marked for style by the authors, and also run against suitable test cases to determine quality.

Results were collected twice. In the first experiment pairs were constructed (wherever possible) of students whom we identified as 'opposites' in terms of the attitude questionnaire. In the second analysis we placed students in pairs with others whom we had identified as 'the same'. Students who came out in the 'middle' group on the questionnaire were always assigned each other as partners.

Our suspicion when we set up the experiment was that students in the code-a-phobe and middle groups would enjoy this experience and perform better than might have been expected on the programming problem, but that students in the 'code-warrior' group would not enjoy working either with other warriors or with other kinds of students.

6 The Results

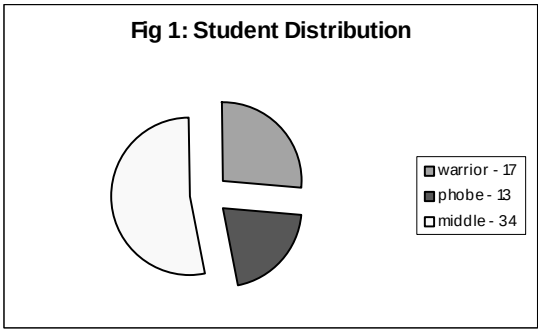
Although the class with which we performed this experiment was quite large (more than 60 students), once the students were divided into pairs and further divided by attitude, groups were too small to gain much of statistical significance. This study can only be a first step in a larger study – but given that proviso there are several interesting things to be gleaned from the data collected.

6.1 Distribution of Code-Warriors and Code-a-phobes

We determined the three groups by the self-placement of the students as expressed on the 'Programming Attitude Questionnaire'. Students who rated themselves as 7-9 were in one group (we will refer to them as 'warriors' in this paper – but did not use this emotive terminology when setting up the student pairs). Students who placed themselves as 1-3 were in another group (in this paper we will refer to them as 'phobes'). The remaining students were in the third or 'middle' group.

---

most of our students, who are considerably less mature and not used to cooperative endeavour.



Again the question of ability *versus* confidence arises. In the determination of the staff *all* these students had reasonable programming experience (in almost all cases, A-level computing which is approximately equivalent to an introductory overview course at most North American Universities). Even so, many of them seem to have little confidence in their own ability[2].

6.2 Self-rating as a prediction of success

We correlated how the students rated themselves with their final marks in the module and the results are illustrated in Figures 2 through 4.

The average mark for the class as a whole and for the middle group was 59.7%. The warrior group averaged 62.8% and the phobe group 55.7%. We performed a t-test on these values and the most significant was the result for the phobe group (p=.143). This is indicative but not statistically significant. The marks for the warrior group were not statistically significantly higher than for the class as a whole and do not come close to a normal distribution as can be seen in Figure 2 (so the t-test is not suitable in any event).

The students who rated themselves as 'warriors' but did not perform well in the module form an interesting group that needs consideration.

6.3 Results after the first ('opposite') experience

After the first experience we asked students questions (based on Laurie Williams' questionnaire) about their enjoyment and performance on the task. In this experiment warriors were placed with phobes when possible and with middles when not[3].

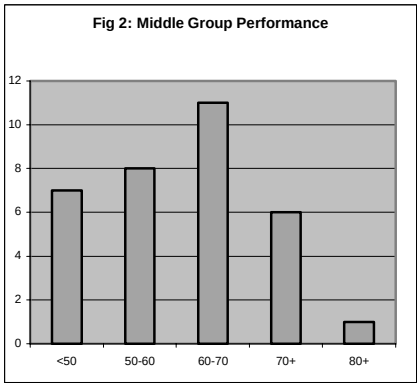
1. Did you enjoy the pair programming experience more than working alone?
   Yes / No / Undecided
   Why / Why Not?
2. Do you think you did a better job with this problem because you solved it in a pair?
   Yes / No / Undecided
   Why / Why Not?
3. Was there anything about the experience you particularly liked?
4. Did you experience any particular frustrations with pair programming?

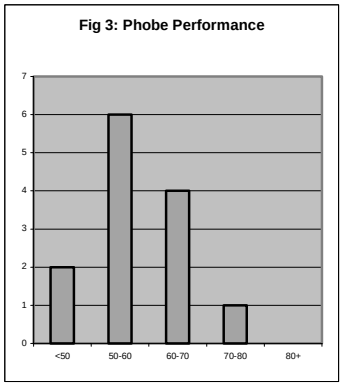
6.3.1 Satisfaction

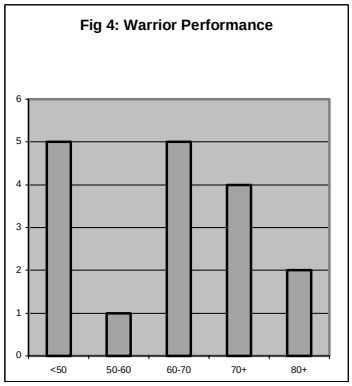
Overall, students enjoyed the experience (66%) and thought that it helped them produce a better solution (66%). Data can be seen in Table 1. Students mentioned that they enjoyed the aspects of

	enjoyment			achievement		
	yes	no	unsure	better	worse	unsure
warrior	9	1	7	8	6	3
phobe	8	2	3	8	2	3
middle	25	1	8	26	2	6
total	42	4	18	42	10	12

Table 1: Results from the First Experience







[2] We think it is fair to say that the stereotype of a code-warrior is male. There were only 7 women in the class, so it is impossible to make generalisations, but one female student rated herself in the phobe group, the others all placed themselves in the middle group.

[3] A few students arrived late and so 2 of the phobes were actually placed with middles.

mutual help, communication and learning new things. The main complaints were about insufficient time for the exercise and the artificiality of the 'switching time' but students also mentioned frustration, guilt and wasted time.

We particularly wished to examine students who were placed with 'opposite' partners in this pairing. The phobic students responded essentially the same as the group as a whole, but when we concentrate on the first row of Table 1 the results tell a different story. Only 53% of the warriors reported enjoyment of the experience and only 47% of them thought that pair programming led to a better solution. This latter result was actually statistically different from that of the rest of the group when compared using the t-test (p=.044).

6.3.2 Performance

When we examine the marks allocated to the various groupings results are also interesting as can be seen in Table 2.

totally opposite pairs- 11	2.23
somewhat opposite pairs - 8	3.19
similar pairs - 13	2.69

Table 2: Grades from First Experience

The highest grades awarded (A was 4.0, B was 3.0 etc.) were in the groups of warriors and middles (and the 2 groups of middles and phobes who achieved an A and a B). The lowest grades awarded were in the warrior/phobe combinations. It could be argued that this was because the less confident students brought down the pair's marks (but see the next section) or that less confident students had less input to the pair's solution – making the solution essentially that of one person. In any case this result bears further investigation.

6.4 Results after the second ('similar') experience

We then repeated the experiment where all the pairs were made up of students from the same group. An extra question was inserted in the questionnaire.

> Did you enjoy this experience more or less than your last pair programming experience?
> More / Less / Same
> Why?

There were 9 groups of warriors, 5 groups of phobes and 14 groups of middles. Unfortunately, 6 of the warriors did not complete the questionnaire (thus conforming to stereotype?)

Results are summarised in Table 3. Overall they were somewhat similar to the first experiment.

Again the warriors were less enthusiastic than the other students, but this time by a smaller percentage; 58% of them enjoyed the experience (as opposed to 64% overall), but 67% thought it led to a better product (as opposed to 65% overall).

Overall 44% of the students *reported* liking the second experience more than the first (and 38% just as much). One explanation could be that they preferred the 'similar' grouping, another that they were just more used to working in a pair. The group that self-reported liking the experience more was the phobes: 60% of whom said they enjoyed this experience more, and none less, than last time. Of the warriors approximately equal numbers reported liking it 'more' and 'less' than the previous time. The change in pairings appears to have had little effect on the warrior group's self-reporting in answering this question (although more of them 'enjoyed' the experience as mentioned in the previous paragraph).

This problem was easier than the one used in the first experiment, as can be seen in Table 4, and results were essentially the same in all groups. There is no evidence of

warrior pairs	3.56
phobe pairs	3.40
middle pairs	3.47

Table 4: Grades from Second Experience

the weaker students (even when there were two of them) producing a weaker result.

7 Conclusions

We draw several conclusions from this experiment, all of which require further examination. First of all with respect to the self-placement on the warrior-phobe scale:

1. Students with very similar background and previous attainment place themselves at very different places on the warrior-phobe scale. (In future work we need to use a scale that carefully separates attitude from expected achievement.)

2. This placement is not necessarily 'correct' in terms of their attainment in a first year programming course. In particular the 5 students who saw themselves as warriors, but achieved less than 50% are worth noting.

	enjoyment			**achievement**			**enjoyed compared to last time**		
	yes	no	unsure	better	worse	unsure	more	less	same
warrior	7	0	5	8	4	0	3	4	5
phobe	8	0	2	8	0	2	6	0	4
middle	17	3	8	17	2	8	12	4	9
total	32	3	15	33	6	12	21	8	18

Table 3: Results from the Second Experience

Secondly, when we examine the results as they apply to pair-programming:

3. Overall, students like pair programming and believe that it helps them achieve good solutions.

4. Students with less self-confidence seem to enjoy pair-programming the most[4].

5. The students whom we have identified as 'warriors' like pair programming the least. This was as we suspected given our student profile.

6. There is some evidence that warriors like pair programming even less when they are paired with phobes and that students produce their best work when paired with students of similar, or not very different, levels of confidence.

8 Future Work

This experiment was conducted during the 2001-2002 academic year. In the current year we have continued to inform our first year students about pair programming and they have all used it in at least one closed lab. Informal feedback indicates that it continues to be popular sand useful for most (but not all) students.

This year we have not, however, given the students the self-confidence survey or controlled for pairings. While we feel that we learnt a lot from this experiment we are worried that the terminology used in the survey may itself be harmful, particularly to less confident students.

In addition, as touched on previously in this paper, we wish to more clearly differentiate between achievement and self-confidence before we proceed further.

We intend to revisit this experiment when we are able to iron out these difficulties, probably later in the year.

9 Acknowledgements

Many thanks to all the students and demonstrators of CS12320 in academic year 2001-2002!

[4] An interesting spin off from this experiment is that some students in the second year of our program adopted pair-programming of their own accord for dealing with less confident group project members.

References

[1] Beck, Kent, *Extreme Programming Explained*, Addison-Wesley (2000).

[2] Davis, H.C., Les Carr, Eric Cooke and Su White, Managing Diversity: Experiences Teaching Programming Principles, *Proceedings of the Second Annual Conference of the LTSN Centre for Information and Computer Sciences*, LTSN-ICS (2001).

[3] McCracken, M., V. Almstrum, D. Diaz, M. Guzdial, D. Hagen, Y. Kolikant, C. Laxer, L. Thomas, I. Utting, T. Wilusz, A Multi-National, Multi-Institutional Study of Assessment of Programming Skills of First-year CS Students, *SIGCSE Bulletin* (December 2001).

[4] McDowell, Charlie, Linda Werner, Heather Bullock and Julian Fernald, The Effects of Pair Programming on Performance in an Introductory Programming Course, *Proceedings of the Thirty Third Technical Symposium n Computer Science Education, SIGCSE 2002*, ACM Press (2002).

[5] Ratcliffe, M.B., J. Woodbury and L.A. Thomas, A Pedagogically Driven, Directed Learning Environment, *Proceedings of the Second Annual Conference of the LTSN Centre for Information and Computer Sciences*, LTSN-ICS (2001).

[6] Thomas, L.A., M.B. Ratcliffe, J. Woodbury and E. Jarman, Learning Styles and Performance in the Introductory Programming Sequence, *Proceedings of SIGCSE 2002*, ACM Press (2002).

[7] Weber-Wulff, Debora, Combating the Code Warrior: A Different Sort of Programming Instruction, *Proceedings of ITiCSE 2000*, ACM Press (2000).

[8] Weinberg, G.M., *The Psychology of Computer Programming*. New York,Van Nostrand Reinhold (1971)

[9] Williams, Laurie, Robert R. Kessler, Ward Cunningham and Ron Jeffries, Strengthening the Case for Pair-Programming, *IEEE Software* (July/Aug 2000).

[10] Williams, Laurie and Robert R. Kessler, Experimenting with Industry's "Pair-Programming" Model in the Computer Science Classroom, *Journal of Computer Science Education* (March 2001)