A Lightweight Systematic Literature Review of Studies about the use of Pair Programming to Teach Introductory Programming

Mark Turner, Rumjit Kaur, Pearl Brereton

School of Computing and Mathematics,
Keele University
m.turner@cs.keele.ac.uk

Keywords: POP-I.A: Learning to Program; POP-V.B: literature review; POP-V.B: Case Studies; POP-VI.E. Computer Science Education Research

Abstract

Background: Our research group is developing and documenting procedures for undertaking systematic literature reviews (SLRs) within the software engineering domain. A question that has arisen is whether the procedures are suitable for use by students over a relatively short period of time. A further question is related to the effectiveness of pair programming for undergraduate students.

Aims: The aims of this research are twofold: to investigate the applicability of the SLR process within the constraints of a 13-week master's level project and to aggregate evidence about the effectiveness of pair programming for teaching introductory programming.

Methodology: To address the first aim a case study approach was taken with a single student applying the SLR methodology under the supervision of an expert reviewer (and member of academic staff). The process was adapted to fit the time available. For the second aim, a modified SLR method was used, based around an analysis of a random sample of the included studies.

Results: The case study found that, with certain modifications to the process, it was possible to undertake an SLR within a limited time period and to produce valid results. As a novice researcher was undertaking the process, it was found that training was required in certain aspects in order to effectively undertake the review. In particular, the distinction between conference and journal publications and the relationship between publications and studies were sometimes unclear to the student. In terms of the results of the SLR, 28 publications reporting empirical studies of pair programming were selected for inclusion, of which nine publications were used for data extraction and analysis. The preliminary evidence from the review suggests that pair programming can have a positive impact on pass and retention rates as well as the students' confidence and enjoyment of programming. However, the evidence also indicates that pair programming does not have a *significant* effect on the marks obtained for examinations and assignments.

Conclusions: The preliminary results are positive, both for the use of pair programming in introductory undergraduate programming courses and for the use of the SLR process for master's level project students. The evidence from the SLR is that pair programming can significantly improve the success and retention rates on programming modules, as well as student's confidence in their work. In terms of the process, this study has demonstrated that it is possible for a student to follow guidelines for conducting SLRs and, with only minor modifications, deliver a valuable project report.

1. Introduction

Over the past four years our research group has been investigating the applicability of the evidence-based paradigm, which is frequently used within the medical domain, to software engineering (Kitchenham *et al.*, 2004). A key element of the evidence-based paradigm is the systematic literature review (SLR), which is a method for finding, evaluating and synthesising all of the available evidence related to a particular research question, in an objective and unbiased manner (Kitchenham & Charters, 2007). An SLR is frequently used to summarise all of the available evidence from empirical studies reported in research papers, which are referred to as primary studies, on a particular topic of interest.

An essential element of an SLR is the production of a protocol, which provides a detailed plan of how the SLR will be conducted. A protocol details the research question(s), the search strategy to be used to find primary studies, the criteria for selecting studies, how the data will be extracted from the studies, and how the data will be synthesised. Importantly, the protocol is published and therefore available to other researchers so that the SLR can be replicated. In contrast to ad-hoc literature reviews, an SLR is a rigorous, replicable, method that aims to minimise bias. Kitchenham and Charters (2007) provide guidelines for conducting SLRs on software engineering topics, and the guidelines have been followed by other researchers, including PhD students, to conduct a comprehensive systematic review of the literature (Woodall, 2007; Jefferies *et al.*, 2008; Beecham *et al.*, 2007).

We believe that SLRs are suitable for research students as they provide a proven methodology with a series of steps for a student to follow, and produce a less biased review of the literature than a non-systematic, ad-hoc review. However, one criticism of SLRs is that they can be resource-intensive, particularly in terms of time. For example, Petticrew and Roberts (2005) provide details of SLRs taking between 216 and 2,518 hours, and Allen & Olkin (1999) propose that the time taken to perform an SLR is related to the number of studies found during the initial search. The majority of SLRs are conducted by two or more researchers and therefore the time taken to conduct a review can increase if the SLR is undertaken by a single researcher. However, the SLR process has proven effective when conducted by PhD students, possibly because they have the requisite experience in reading and interpreting research papers (Woodall & Brereton, 2006). However, a question remains as to how applicable the SLR process would be to less experienced students, such as students on taught master's courses or even undergraduate students. Keele University has a taught master's degree in Information Technology and Management, which includes a 13-week individual project. We believed that it would be possible to conduct an SLR for such a project, and that any such project would provide valuable results by providing a rigorous methodology for the student to follow (therefore avoiding some of the difficulties experienced by less experienced students when conducting research). However, 13 weeks is an extremely short time period in which to conduct an SLR and any student undertaking the project would be unlikely to have previous research experience. Therefore, the decision was taken to monitor the progress of the project through a case study approach. This paper outlines the results of the case study, based on monitoring the progress of a student undertaking an SLR for a 13-week project on the topic of pair programming.

2. Pair Programming

Computer programming is a compulsory aspect of the majority of undergraduate computing courses. However, teaching introductory programming to first year undergraduate computing students can be an extremely challenging task, particularly as many first year students have little or no experience of computer programming before commencing their degree course (Dunican, 2002). Indeed, the difficulty experienced when learning programming is often thought to be a contributing factor in the retention rates and levels of student engagement on undergraduate computing courses (Miliszewska *et al.*, 2007).

Programming has traditionally been thought of as a solitary activity and this view has been echoed throughout computing education (Cockburn & Williams, 2001; McDowell *et al.*, 2003). However, attempting to develop the required analytical and problem solving skills in order to learn to program is a very difficult task for first year undergraduates, and is especially difficult when doing it alone (Somervell, 2006). An alternative method of teaching programming is instead to enable two students to work collaboratively on a single computer on the same program or piece of code, a technique known as pair programming (Williams & Upchurch, 2001). The potential benefits of pair programming include increased productivity, knowledge transfer, learning and morale, and fewer defects (Freeman *et al.*, 2003; Haungs, 2001). As a result of the increasing use of pair programming in an academic context, a number of empirical studies have been conducted comparing the effectiveness

– measured either quantitatively through examination or assignment marks or qualitatively, through enjoyment or confidence – of pair programming against individual programming.

Pair programming is one technique being considered as a possible way to teach programming to first year undergraduate students at Keele University. An informal search looking for evidence of the success of pair programming in similar situations found a relatively small number of publications on the topic, particularly publications that were reporting the results of empirical studies. Therefore, it was felt that this was a suitable topic for a student on a master's degree course. This paper outlines the results of the project, concerning both the practicality of the SLR methodology for use in master's level projects and the findings of the SLR itself in relation to the effectiveness of pair programming on undergraduate courses.

3. Study design

The SLR followed the guidelines described in Kitchenham (2004). Therefore, the student began the SLR by developing a protocol (as outlined in section 1). The aim of the SLR was to find and collate all of the available evidence to investigate the following research question: Is pair programming an effective method for teaching introductory programming to undergraduate students?

The following sections describe the SLR, and explain how the process was amended in order to be conducted during a 13-week project.

3. 1. Search strategy

The first stage in developing a set of search terms to use in the SLR was to outline the population, intervention, outcomes, and experimental designs that were of interest, based upon the research question. These were:

Population: Students taking introductory programming modules on undergraduate computing courses.

Intervention: Pair programming.

Outcomes of relevance: Measures of the 'effectiveness' of pair programming in terms of student performance and attitude against students that were not paired.

Experimental design: Empirical studies.

As outlined by Kitchenham (2004), the details of the population, intervention, outcomes, and experimental design were used to define the search terms. Firstly, the major terms were extracted from the definitions. Secondly, alternative spellings and synonyms were identified for each of the major terms, and these were combined through the use of Boolean OR and AND.

Due to the time scale involved, a decision was made to only search electronic resources for the SLR and not to conduct manual searching of journal or conference proceedings. Also, as the topic was related to computing, the search was restricted to four digital libraries within the domain (ACM Portal, IEEE Explore, Science Direct and Web of Science). A difficulty encountered was that the search string needed to be amended in order to meet the needs of the different interfaces of each digital library (Brereton *et al*., 2007). Table 1 shows the search strings used for each of the digital libraries searched.

Database	Search String
ACM Portal	“Pair programming” AND (empirical OR introductory OR undergraduate)
IEEExplore	(‘Pair programming’ <and> (empirical <or> introductory <or> undergraduate) <in> (pdfdata, metadata))
Science Direct	“Pair programming” AND empirical OR introductory OR undergraduate
Web of Science	Pair programming* AND undergraduate

Table 1 – Digital libraries and associated search strings

Web of Science in particular appeared to interpret the Boolean search strings unexpectedly, and using the full string (as used for ACM Portal) resulted in the retrieval of many irrelevant publications. The simplified search string, as shown in Table 1, was finally used for the Web of Science digital library in order to decrease the number of irrelevant results.

The search strings were validated by their ability to locate previously known publications on the topic, particularly Williams *et al*. (2003).

3. 2. Selection criteria

The full list of publications identified by the searches was initially subject to a review of the title, keyword, and abstract. Any publications that seemed irrelevant during this initial screening process were excluded. Full copies were obtained of the remaining publications, which were then evaluated against a set of inclusion and exclusion criteria. Where there was uncertainty regarding a particular publication, the student passed a copy of the full publication to the supervisor or another academic with experience of SLRs. The result was a list of publications that the student felt should be included in the SLR.

The following sub-sections detail the inclusion and exclusion criteria used and the process by which the criteria were applied to the set of publications.

3.2.1. Inclusion criteria

The following inclusion criteria were applied to all of the publications that were not excluded by the first screening process. Any publications that described the following were included in the SLR:

- Empirical studies concerning the use of pair programming in the teaching of introductory programming for undergraduate students.
- Primary studies that provided data about the effectiveness (in the form of examination or assignment marks, retention rates, measures of assignment quality, or student’s enjoyment, attitude, or confidence) of pair programming against solo programming
- Empirical studies consisting of more than two participants.

In addition, it was decided that if data relating to the same primary study was reported in several publications, only the most recent would be included in the SLR. Also, any publications that reported the results of several primary studies would be examined as separate studies for the data extraction and analysis.

3.2.3. Exclusion criteria

The publications were also assessed against the following criteria, and any that met the criteria were excluded from the review:

- Publications that only provide a summary of a relevant empirical study.
- Publications describing primary studies that are based upon pair programming, but do not investigate the pedagogical aspect or do not provide measures of effectiveness.
- Publications describing or measuring Extreme Programming practices, but which have little or no results concerning pair programming.
- Publications that provide incomplete results.
- Publications that are only available in the form of an abstract or Powerpoint presentation.

Any publications that report incomplete results should ordinarily be included in an SLR if they meet the inclusion criteria, as attempts may then be made to contact the authors of the publication to gain access to the full data set(s). However, as there was a strict time limit of 13 weeks for this SLR, any such publications were excluded.

3. 3. Quality assessment

Once the inclusion/exclusion criteria have been applied, the next stage in an SLR according to Kitchenham (2004) is to assess the quality of all remaining publications. The aim is to assess the validity of the primary studies described in the publications and to highlight any that it may be important to remove from the synthesis when performing sensitivity analyses. However, due to the timescale involved and the limited experience of the student regarding empirical studies and statistical techniques, it was decided to perform a basic quality assessment of the included studies but not to use this as a basis for sensitivity analyses. This provided the student with experience regarding the technique and of assessing the quality of empirical studies, without needing to understand sensitivity analysis techniques or the risk of biasing the study if an incorrect assessment was made. The following criteria were applied to each of the publications to assess the quality:

- Does the publication clearly describe the experimental design and methods used in the study?
- Is the study research design valid and consistent with the characteristics of the data given?
- Are the findings from the studies clearly documented and complete?
- Is there justification for any missing data or inconsistency?
- Are the sample size and the population clearly defined?
- Is information provided on the settings in which the study was conducted?

3. 4. Data extraction strategy

Due to the timescale of the SLR, it was decided that the data extraction and synthesis should concentrate on a subset of the included publications, rather than attempting to analyse the data from all of the publications. However, in order to minimise the risk of bias, random sampling was used to select the subset of the included publications for extraction and analysis. The main reason for using a subset of the total included publications was to enable the student to gain experience in the whole SLR process and to reduce the risk of the project not being completed on time.

An electronic form was used to store the data extracted from each of the publications in the subset. In order to assess the validity of the data extraction form, it was piloted before undertaking the main review by extracting data from a known publication (Williams *et al.*, 2003). The form was then changed according to the results of the pilot, and subsequently went through several iterations and changes based on the way that the data was reported in the publications (see Section 5).

One extraction form was completed for each primary study. Therefore, if for example one publication contained data on two primary studies, two extraction forms would be completed. This was because the data synthesis would be on the basis of studies rather than publications (see Section 4). The extraction form concentrated on basic details of each primary study, such as the authors, the empirical method employed, and the sample size, along with the results of the study in terms of the pass rates, assignment marks, assignment quality, retention rates, attitude, confidence and enjoyment for both paired and non-paired students.

3. 5. Data synthesis

The extracted data is summarised using Table B1 and Table B2 in Appendix B. Table B1 summarises the results of each individual primary study, whilst Table B2 details frequency counts in relation to the number of primary studies that found pair programming effective, for each of the different measures. The measures of effectiveness were determined from the measures reported in the subset of publications that were selected for data extraction and analysis. The results were synthesised according to eight effectiveness measures: examination marks, assignment marks, assignment quality, pass rates, retention rates, confidence, enjoyment, and attitude.

4. Study results

The preliminary results of the SLR are summarized in the following sections.

4.1. Included Publications

The number of publications that were returned by each digital library, along with the number of publications included in the review after applying the inclusion and exclusion criteria, are shown in Table 2. The number of included publications in Table 2 refers to the number of unique publications selected for inclusion from each digital library and therefore does not indicate if publications were found in multiple digital libraries.

Digital Library	Total Number of publications returned	After initial screening process I – Included U – Unclear E - Excluded			Publications selected for inclusion after further analysis of unclear publications
		I	U	E	
ACM Portal	200	11	13	176	14
IEEExplore	68	10	10	48	10
ScienceDirect	31	3	1	27	4
Web of Science	6	4	1	1	4
Total	305	28	25	252	32

Table 2 - Search Results

The initial screening process was based on an analysis of the titles and abstracts for all publications returned by the digital libraries, with publications being included, excluded, or marked as unclear. As indicated in Table 2, at this stage almost as many publications were marked as unclear as were included in the review (25 and 28, respectively). This could be a direct result of the quality of abstracts within software engineering research, which makes it difficult for experienced researchers when conducting SLRs and which will be exacerbated when a novice researcher is conducting the screening process (Brereton *et al.*, 2007). Full texts were obtained for all of the publications marked 'unclear' and after further analysis four of the 25 publications were selected for inclusion. This resulted in the selection of 32 publications for inclusion in the review.

However, after further analysis it was discovered that five of these 32 publications were essentially reporting the results of the *same* primary study. As the results had been reported slightly differently in each of the five publications, the student was not sure if they were the same. A further reading indicated that they were all based on the same primary study and were reporting the same results and so four of the five publications were excluded, resulting in 28 publications being included in the final SLR (see section 5).

As described in section 3.4, in order for the review to be conducted within the confines of a 13-week project, the decision was made to take a random sample of the included publications and to use this subset for data extraction and analysis. The titles of the 28 publications were allocated numbers and then randomised using a spreadsheet package. Every 3rd publication was selected from the list, resulting in nine publications being randomly selected, plus the one publication that was randomly selected and used as a pilot for the data extraction form (see section 3.4).

It should be noted that one of the ten publications in the randomly selected subset was found to be irrelevant to the SLR after the analysis had begun (Hanks, 2005). This was because the study was empirical but was found to be comparing two different uses of pair programming and was not comparing pair programming against solo programming. As a result, nine publications were used for the analysis.

4.2. Analysis of pair programming

It was not possible to conduct a statistical meta-analysis due to the diversity in the study types, the variables used by each study, and the fact that certain variables were qualitatively reported rather than quantitatively (i.e. measures of *enjoyment* or *attitude towards programming*). Therefore, the analysis of the results was based upon frequency counts, proportions, and binomial proportion confidence intervals. Proportions (p) are calculated using the following formula:

$$p = e/(e + ne)$$

where *e* refers to the number of primary studies reporting a positive effect in relation to the dependent variable and *ne* refers to the number of studies reporting that no significant effect was found. The following equation was used to calculate the 95% confidence interval of the proportions.

$$p + 1.96 \sqrt{} p\ (1-p / n)$$

The results were analysed in relation to eight indicators of ‘effectiveness’, based on the results presented in the nine randomly selected publications for which data was extracted. One publication included data relating to two primary studies, whereas all others reported on a single primary study only (Williams *et al.*, 2003). The results in this section therefore are in terms of the number of *primary studies* rather than the number of *publications*, and a total of 10 primary studies were analysed which were described in nine publications. As the results are only based on a subset of the total publications selected for inclusion, the results presented here are preliminary and a follow up SLR is being conducted to extract the data from all of the included studies.

Table 3 illustrates the results of the studies that reported the effectiveness of the pair programming technique in relation to students’ marks in examinations.

	Number of Primary studies *(n)*	Proportion	95% CI (1.96)
Effective *(e)*	1	0.2	-0.15 - 0.55
Not Effective *(ne)*	4	0.8	0.53 – 1.13

Table 3 – Primary studies using examination marks as dependent variable

Five primary studies used exam marks as a measure of the success of pair programming (McDowell *et al.*, 2003; Nagappan *et al.*, 2003; Williams *et al.*, 2003; Mendes *et al.*, 2006). Mendes *et al.* (2006) was the only study reporting a statistically significant difference (p=0.003) between the marks obtained by students who used pair programming and students who programmed independently. The majority of the studies (0.8) found that, whilst pair programming had no significant effect on the students' examination marks it did not have any negative impact. Therefore, the studies found that whilst pair programming did not significantly *increase* students' marks in examinations it also did not significantly *decrease* their marks.

	Primary studies *(n)*	Proportion	95% CI (1.96)
Effective *(e)*	2	0.4	-0.03 – 0.83
Not Effective *(ne)*	3	0.6	0.17 – 1.03

Table 4 - Number of primary studies using assignment marks as dependent variable

A similar result was found for the effect of pair programming on students' marks on assignments (see Table 4). Five primary studies were included that reported assignment marks as a dependent variable (Mendes *et al.*, 2006; Nagappan *et al.*, 2003; Williams *et al.*, 2003; Wilson *et al.*, 1993), and of those only two studies reported any significant positive effect (Mendes *et al.*, 2006; Williams *et al.*, 2003). Similar to the results for examination marks, the indication from the studies counted as 'not effective' in Table 4 is that the marks for students who programmed in pairs were not significantly different, either lower or higher, than those who programmed independently.

	Primary studies (n)	Proportion
Effective (e)	5	1
Not Effective (ne)	0	0

Table 5 – Number of primary studies using assignment quality as dependent variable

As shown by Table 5, all five primary studies that used assignment quality as a measure of effectiveness found that the quality of assignments that were completed by paired students was 'better' in terms of a particular measure than those completed by individuals (Bipp *et al.*, 2007; McDowell *et al.*, 2003; Xu & Rajlich, 2006; Wilson *et al.*, 1993; VanDeGrift, 2004). The majority of primary studies reporting assignment quality were different studies to those reporting the marks on assignments, with the exception of Wilson *et al.* (1993). This particular study reported positive results for assignment quality for the paired students, but no statistically significant difference in the actual marks for the assignments (Wilson *et al.*, 1993). In this study, assignment quality was measured in terms of the readability of the code and the "degree to which problem solvers can articulate proposed solutions". Therefore, it is possible that the quality of assignments could be higher for students who were paired, but depending on how the assignments were marked students who paired did not get significantly higher marks than students who programmed individually. An interesting comparison would be possible if the other studies that reported assignment quality also reported the marks given for the assignments.

	Primary studies *(n)*	Proportion	95% CI (1.96)
Effective	4	0.8	0.45 – 1.15
Not Effective	1	0.2	-0.15 – 0.55

Table 6 – Number of primary studies using pass/success rates as dependent variable

Five primary studies measured the effect of pair programming on the pass or success rates of students on programming modules or courses (Williams *et al.*, 2003; McDowell *et al.*, 2003; Nagappan *et al.*, 2003; Mendes *et al.*, 2006). The majority of the studies (4/5) found that pair programming was effective in improving the pass rates for undergraduate programming modules, with the exception of Nagappan *et al.* (2003) which only found that pair programming had an effect on non-Computer Science majors and had no significant effect on the CS majors.

	Primary studies *(n)*	Proportion
Effective	4	1
Not Effective	0	0

Table 7 – Number of primary studies using retention as a dependent variable

All of the four primary studies that included student retention as a measure of effectiveness found that pair programming significantly improved the retention rates within programming courses (McDowell *et al.*, 2003; Williams *et al.*, 2003; Mendes *et al.*, 2006). For example, McDowell *et al.* (2003) found students who work in pairs are most likely to complete the course (90.8%) as opposed to students working independently (80.4%). Williams *et al.* (2003) found students who have participated in a pairing program were more likely to pursue computing related courses further. All of the studies which included the measure found that pair programming had been particularly effective in retaining students on computing courses and, in some cases, increased the number going on to pursue programming further. Whilst this review found that pair programming was effective in increasing the retention rates of courses, Williams (2000) states that “pair pressure” may influence the results, whereby students may stay on the course due to pressure from their paired partner.

	Primary studies *(n)*	Proportion
Confidence		
Effective	4	1
Not Effective	0	0
Enjoyment		
Effective	5	1
Not Effective	0	0
Attitude		
Effective	4	1
Not Effective	0	0

Table 8 – Number of primary studies using confidence, enjoyment and attitude as dependent variables

All of the primary studies that were analysed found that student’s confidence, enjoyment and attitude towards programming improved with the introduction of pair programming (Table 8). Four studies used the students’ confidence in programming as a measure (McDowell *et al.*, 2003; Wilson *et al.*, 1993; VanDeGrift, 2004; Slaten *et al.*, 1993). McDowell *et al.* (2003) suggests that students who program in pairs acquire greater confidence in programming and their solutions (89.4% versus 71.2%, significance level $p<0.001$). However, one of the studies that used confidence as a measure was based purely on interviews with six students (Slaten *et al.*, 1993). Even removing this study, the remaining three studies all reported significant positive differences in confidence levels between those who programmed in pairs and those who programmed individually.

Five primary studies measured students' enjoyment of programming exercises (McDowell *et al.*, 2003; Williams *et al.*, 2003; Wilson *et al.*, 1993; Mendes *et al.*, 2006). Wilson *et al.* (1993) reports that collaboration helps novice users grasp key concepts and improve their problem solving skills, and increases their enjoyment. However, the measurement of enjoyment in the studies differs greatly. For example, one study simply measured students' level of enjoyment of pair-programming, without comparing enjoyment of programming between paired and solo (Mendes *et al.*, 2006), while another compared enjoyment of the task and provided comparative data between solo and pairs (Wilson *et al.*, 1993). Four studies used attitude as a measure and all four reported positive results (Nagappan *et al.*, 1993; Williams *et al.*, 2003; VanDeGrift, 2004). However, similar to the measures for enjoyment, the studies used differing measures.

5. Discussion of the SLR process

The first issue encountered with a master's level student undertaking an SLR was a misunderstanding of the nature of the process and the steps involved in conducting such a review. This meant that the development of the protocol took longer than expected and required many changes and revisions. The initial development of the protocol was based on the first set of SLR guidelines outlined in Kitchenham (2004). However, during the project a new version of the SLR guidelines was produced that included more examples that were designed to help students to understand the process (Kitchenham & Charters, 2007). Therefore, it is recommended that students undertaking a SLR use the guidelines outlined in Kitchenham & Charters (2007). However, the guidelines still require a certain level of statistical knowledge on the part of the researcher, particularly with regard to the type of statistics to use when conducting a meta-analysis, which a master's level student is unlikely to possess. Therefore, we found that training in certain basic statistical techniques was required, such as proportions, confidence intervals, and significance levels.

The issue of a protocol requiring continuous revision, particularly when being developed by a single novice researcher, has been highlighted previously. Woodall & Brereton (2006) suggested that "step wise refinement" of the protocol is required, accompanied by guidance from the supervisor. This is exacerbated when the student is not an expert in the subject of the SLR. Despite piloting the protocol, it still required considerable modifications after the search strategy had been applied. After examining the studies in detail, it was found that several studies reported different dependent variables than expected, and many reported the results in different ways. This meant that the data extraction form needed to be amended. A record was kept of all of the divergences from the initial version of the protocol. This is a problem that faces experienced researchers as well as novices (Staples & Niazi, 2003). Staples and Niazi (2003) suggest that a data management resource would be highly beneficial in ensuring consistency among the phases of the SLR when multiple changes to documents are required, such as protocols, and this is particularly important for novice researchers.

A further issue encountered was with the definition of the inclusion/exclusion criteria. Due to a lack of knowledge about the domain and lack of experience in conducting SLRs, the initial inclusion/exclusion criteria was relatively broad and did not concentrate on the *effectiveness* of pair programming. This meant that when conducting the screening process, studies were included that discussed issues relating to the compatibility of pairs rather than evaluating the technique. A further issue was how effectiveness was defined, and in order to ensure that the review could be undertaken in the time period, it was necessary to impose restrictions on exactly what measures of effectiveness would be included. This resulted in several iterations of the screening process and changing the inclusion/exclusion criteria accordingly. Therefore it is recommended that supervisors work with the student to define the inclusion/exclusion criteria to tighten them in order to avoid later revisions.

The reviewer also encountered difficulty in defining and applying the inclusion/exclusion criteria for the SLR. In order to address the research question and to assess the effectiveness of pair programming, studies needed to be included that compare pair programming against other methods for teaching programming at an undergraduate level. Whilst this was implicit when discussing the requirements of the review, it was not explicitly stated in the inclusion criteria in the first version of the protocol. This resulted in the student including one study which compared two different applications of pair programming, rather than comparing pair programming against individual programming (Hanks, 2005). Unfortunately, this was also randomly chosen as one of the studies to analyse and therefore some of the data needed to be amended after analysis had begun. The inclusion criteria were then amended to include the phrase *"against solo programming*". It is recommended that, wherever possible, supervisors check *all* of the publications that are selected for analysis against the inclusion/exclusion criteria rather than checking a sample. As this SLR was only using a subset of ten publications it was possible to achieve this but in a larger SLR this may not be possible due to time constraints. The student also found that in many cases it was very difficult to determine if a publication was relevant or not by reading the title and abstract alone, which agrees with what other reviewers in software engineering have experienced (Brereton *et al.*, 2007; Budgen *et al.*, 2007; Staples & Niazi, 2006). This fact can significantly increase the time needed to conduct an SLR and it was another factor in deciding to only synthesise and extract a selection of publications.

A further difficulty the reviewer faced was differentiating between *studies* and *publications*. It was found that effectively one study had been reported in five different publications, and due to lack of experience in reading academic publications, particularly relating to empirical work, the reviewer included all five publications as different studies. This was discovered when the data extraction forms were reviewed by other researchers and resulted in only one of the publications being included. Therefore, it is recommended that novice researchers may require some training in reading academic papers, and in the different types of empirical study, before undertaking a review. This would also help to address another issue encountered – that of confusing conference and journal publications – as occurred during this SLR. It also indicates that when a novice researcher is conducting a review, the protocol should be explicit in how to deal with the issue of a) multiple studies reported in one publication and, as in this case, b) multiple publications reporting the same study.

The student also encountered difficulties with performing an in-depth quality assessment of the publications. This was partly due to a lack of experience with the way that empirical studies are reported in academic publications, and therefore it was difficult to make a decision regarding the quality of studies even when following pre-existing quality criteria as in Kitchenham (2004). Kitchenham and Charters (2007) provides further details regarding quality assessment in SLRs, but it is still difficult for a novice reviewer to interpret. Due to the differences in empirical studies it is often necessary for reviewers to develop their own guidelines for specific reviews, which is very difficult for a novice. As SLRs become more prevalent in software engineering it is likely that other quality assessment guidelines will be produced, which may help novices by allowing them the ability to choose different criteria that are relevant to their reviews.

Finally, it was necessary to reduce the scope of the SLR to fit in with the timescale of a master's level project and to accommodate a single researcher. This was achieved by allowing the student to go through all of the steps of the SLR but only performing the extraction and analysis on a sample of the included publications. This allowed the student to gain experience in the whole SLR process, and to produce valuable results, but without attempting to extract and analyse the data from all of the 28 included publications, which can be the most time-consuming element of the SLR process (Petticrew & Roberts, 2005). The number of publications sampled will depend upon the research questions and the topic, and in certain cases it would be possible for the student to extract data from all of the publications if the topic was sufficiently narrow. It was found with this SLR that ten publications were appropriate for a timescale of 13 weeks.

6. Conclusions

Overall, there are two important results from this study. Firstly, the evidence from 10 empirical studies synthesised as part of an SLR is that whilst the use of pair programming with undergraduate students may have no *significant* impact on marks (in both assignments and examinations) it can improve the pass and retention rates on programming modules, as well as the students confidence in their work and their attitude towards programming.

Secondly, the study was also a test of the systematic literature review method. Traditionally, the view is that SLRs can take a great deal of time and are therefore resource heavy in terms of time and reviewer effort. This study found that it is possible to go through all of the steps of an SLR, from formulation of the research question(s) and creation of a protocol to the synthesis of the results, within 13 weeks. This was achieved by doing a full review up to the selection of studies to include, and then taking a random sample of studies on which to perform the data extraction and synthesis. This will be different for other reviews, and will depend on the number of studies initially found during the searches (as if a great deal of results are returned, the initial screening process can take longer than planned) and the number of studies selected for inclusion. It was found that for our SLR, where 28 publications were selected for inclusion, it was appropriate to select every 3rd publication – or roughly 10 publications. This proved to be an appropriate number for a 13-week project being undertaken by a master's level student.

However, this is also the primary limitation of this SLR with regard to the findings. The results of the SLR only relate to a sample of the publications selected for inclusion in the SLR (nine out of 28). In order to reduce the bias, a random selection process was defined to select which of the 28 publications would be used for the data extraction and synthesis. There is the possibility that the results of the remaining 18 publications will provide extremely different results to those nine selected. In particular, they may provide further information to explain the apparent anomaly that pair programming has been shown to have no significant effect on marks but a significant positive effect on pass rates. Publication bias may also be a factor as the databases searched indexed only papers selected for publication and did not index grey literature (such as technical reports, and PhD theses). Therefore, the issue of publication bias, whereby published studies may be concentrating purely on positive results with any negative results published via technical reports or other means, may have had an effect on the results. A further issue is that it was not possible to conduct a detailed quality assessment and sensitivity analysis.

Further work is planned to complete the SLR by extracting and synthesising the results of the remaining 18 publications selected for inclusion in the review. The results of the full SLR can then be compared with the results from the analysis of the subset in order to determine if the results presented here are representative of the review as a whole. A further study is planned to replicate the review by an experienced researcher, to determine if a researcher experienced in empirical studies would create the same search strings and select the same studies as the master's student. Again, the results of the review conducted by an experienced researcher can then be compared with the preliminary results produced by the master's student.

6. Acknowledgements

We thank the members of the EPSRC-funded Evidence-based Practices Informing Computing (EPIC) project for their continuing research into the use of evidence-based practices within software engineering and for their useful discussions relating to this work.

7. References

Allen, I, and Olkin, I. (1999) Estimating time to conduct a meta-analysis from number of citations received, Journal of the American Medical Association, 282(7), 634-5.

Allison, I., Orton, P. and Powell. H. (2002). A virtual learning environment for introductory programming, In Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences, 48-52.

Beecham, S., Baddoo, N., Hall, T., Robinson, H. and Sharp, H. (2007) Motivation in Software Engineering: A Systematic Literature Review, University of Hertfordshire, Technical Report 464

Brereton, O. P., Kitchenham, B., Budgen, Turner, M. and Khalil, M. (2007) Lessons from applying the Systematic Literature Review process within the Software Engineering domain, Journal of Systems & Software, 80(4), 571–583.

Budgen, D., Kitchenham, B., Charters, S., Turner, M., Brereton, P. and Linkman, S. (2007) Preliminary results of a study of the completeness and clarity of structured abstracts, in Proceedings of EASE 2007, BCS-eWiC, 64-72.

Cockburn, A. and Williams, L. (2001). The costs and benefits of Pair Programming, Addison Wesley.

Dunican, E. (2002) Making the Analogy: Alternative delivery techniques for first year programming courses. In J. Kuljis, L. Baldwin & R. Scoble (Eds.), Proceedings from the 14th Workshop of the Psychology of Programming Interest Group, Brunel University, 89-89.

Freeman, S., Jaeger, B. and Brougham, J. (2003) Pair Programming: More Learning and Less Anxiety in a First Programming Course, In Proceedings of the ASEE Annual Conference.

Haungs, J. (2001) Pair programming on the C3 project, IEEE Computer, 34(2), 118-119.

Jefferies, C., Brereton, P., Turner, M. (2008), A Systematic Literature Review of Approaches to Reengineering for Multi-Channel Access, in Proceedings of the 12th International Conference on Software Maintenance and Reengineering (CSMR 2008), pp. 258-262.

Kitchenham, B. (2004) Procedures for Performing Systematic Reviews, Joint Technical Report Software Engineering Group, School of Computing and Mathematics, Keele University, UK and Empirical Software Engineering, National ICT Australia Ltd, Australia.

Kitchenham, B. and Charters, S. (2007) Guidelines for performing Systematic Literature Reviews in Software Engineering, Version 2.3, EBSE Technical Report (EBSE-2007-01).

Kitchenham, B.A. Dybå, T. and Jørgenson, M. (2004). Evidence-Based Software Engineering, in *Proceedings of ICSE 2004*, IEEE Computer Society Press, 273-281.

McDowell, C. Hanks, B. and Werner, L. (2003) Experimenting with Pair Programming in the Classroom, ACM SIGCSE Bulletin, Proceedings of the 8th annual conference on Innovation and technology in computer science education (ITiCSE '03), 35(3), 60 – 64.

Miliszewska, I. and Tan, G. (2007). Befriending Computer Programming: A Proposed Approach to Teaching Introductory Programming, Journal of Information Technology Education: Issues in Informing Science and Information Technology, 4, 278-289.

Petticrew, M. and Roberts, H. (2005). Systematic reviews in the social sciences: A practical guide, Blackwell.

Somervell, J. (2006) Pair programming: Not for everyone?, International Conference on Frontiers in Education: Computer Science and Computer Engineering, 303-307.

Staples, M. and Niazi, M. (2006) Experiences Using Systematic Review Guidelines, In Proceedings of the 10th International Conference on Evaluation and Assessment in Software Engineering (EASE 2006), BCS-eWiC, 79-88.

Williams, L. (2000), Strengthening the case for pair programming, IEEE Software, 17(4), 19-25.

Williams, L. and Kessler, R. (2000) The effects of "Pair-Pressure" and "Pair-Learning" on Software Engineering Education, In Proceedings of the 13th Conference on Software Engineering Education & Training, IEEE Computer Society Press, 59 – 65.

Williams, L. and Upchurch, R. (2001) In Support of Student Pair Programming, SIGCSE Conference on Computer Science Education, 327-331.

Williams, L. McDowell, C., Nagappan, N., Fernald, J. and Werner, L. (2003) Building Pair Programming Knowledge through a Family of Experiments, In Proceedings of the 2003 international Symposium on Empirical Software Engineering, 143 – 152.

Woodall, P. and Brereton, P. (2006), Conducting a systematic literature review from the perspective of a PhD student, In Proceedings of the 10th International Conference on Evaluation and Assessment in Software Engineering (EASE '06), BCS-eWiC, 138.

Woodall, P. (2007) Controlling inference in service-based systems, PhD Thesis, Keele University.

Appendix

A. Studies selected for analysis

Bipp, T. Lepper, A. and Schmedding, D. (2008) Pair Programming in Software Development Teams: An Empirical Study of its Benefits, Information and Software Technology, 50(3), 231-240.

Hanks, B. (2005) Student performance in CS1 with distributed pair programming, In Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education (ITiCSE '05), 37(3), 316-320.

McDowell, C. Werner, L., Bullock, H. and Fernald, J. (2003) The impact of pair programming on student performance, perception and persistence, In Proceedings of the 25th International Conference on Software Engineering (ICSE '03), 602-607.

Mendes, E., Fakhri, L., and Luxton-Reilly, A. (2006) A replicated experiment of pair-programming in a 2nd-year software development and design computer science course, ACM SIGCSE Bulletin, In Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education (ITICSE '06), 38(3), 108-112.

Nagappan, N., Williams, L., Ferzli, M., Wiebe, E., Yang, K., Miller, C. and Balik, S. (2003) Improving the CS1 experience with pair programming, ACM SIGCSE Bulletin, In Proceedings of the 34th SIGCSE technical symposium on computer science education (SIGCSE '03), 35(1), 359-362.

Slaten, K.M., Droujkova, M., Berenson, S.B., Williams, L. and Layman, L. (2005) Undergraduate student perceptions of pair programming and agile software methodologies: Verifying a model of social interaction, In Proceedings of the Agile Conference, 323 – 330.

VanDeGrift, T. (2004) Coupling pair programming and writing: learning about students' perceptions and processes, ACM SIGCSE Bulletin, 36(1), 2-6.

Williams, L., McDowell, C., Nagappan, N., Fernald, J. and Werner, L. (2003) Building Pair Programming Knowledge through a Family of Experiments, In Proceedings of the 2003 International Symposium on Empirical Software Engineering (ISESE '03), 143-152.

Wilson, J., Hoskin, N. and Nosek, J. (1993). The benefits of collaboration for student programmers, ACM SIGCSE Bulletin, 160-164.

Xu, S. and Rajlich, V. (2006) Empirical Validation of Test-Driven Pair Programming in Game Development, In Proceedings of the 5th IEEE/ACIS International Conference on Computer and Information Science, 500–505

B. Data Synthesis tables

Study ID	Author	No. of pairs	% Pairs passed	No. of Solo	% Solo Passed	Statistical sig.	Effectiveness found? Y/N)	Incomplete data? (Y/N)

Table B1. Individual Study Summary Results

Effectiveness of the pair programming tool	No. of studies
Effective	
Not effective	
Total no. of studies investigated	

Table B2. Study frequency counts