

UNIVERSITY OF KWAZULU-NATAL

**COOPERATIVE LEARNING TECHNIQUE FOR
TEACHING OBJECT ORIENTED PROGRAMMING**

By

**Thamotharan Prinavin Govender
891288849**

A dissertation submitted in fulfilment of the requirements for the degree of

Master of Education

**School of Science, Mathematics and Technology Education
Faculty of Education**

Supervisor: Dr D.W Govender

November 2013

PREFACE

Two papers emanated from this study. These papers were presented at the Durban University of Technology, Faculty of accounting and Informatics Research Day, October 2010, and at the SAICSIT postgraduate symposium, October 2011 entitled “Back to basics find a pair and program”. The papers focused on the role of pair programming as a teaching strategy and the advantages and disadvantages of implementing pair programming.

DECLARATION

This study is the original work of the author and has not otherwise been submitted in any form for any degree or diploma to any University. Where use has been made of the work of others, such has been duly acknowledged in the text.

Signature (T.P Govender)

Date

Place

TABLE OF CONTENTS

PREFACE	i
DECLARATION	ii
LIST OF TABLES	vi
LIST OF FIGURES	vii
GLOSSARY OF TERMS	viii
LIST OF ABBREVIATIONS	ix
ABSTRACT	x
ACKNOWLEDGEMENTS	xi
CHAPTER ONE: INTRODUCTION	1
1.1 INTRODUCTION	1
1.2 PURPOSE OF THE STUDY	2
1.3 PROBLEM STATEMENT	2
1.4 RATIONALE FOR THE STUDY	2
1.5 OBJECTIVE OF THE STUDY	4
1.6 THEORETICAL AND CONCEPTUAL FRAMEWORK	4
1.7 KEY RESEARCH QUESTIONS	5
1.8 RESEARCH DESIGN AND METHODOLOGY	5
1.9 STRUCTURE OF THE STUDY	7
CHAPTER TWO: LITERATURE REVIEW	8
2.1. INTRODUCTION	8
2.2 DIFFERENCE BETWEEN COLLABORATIVE LEARNING, COOPERATIVE LEARNING, PP AND CAP PROGRAMMING	10
2.3 HISTORY OF PP	11
2.4 SHORTCOMINGS OF PP	11
2.5 FIVE PRINCIPLES OF COOPERATIVE LEARNING	14
2.6 BENEFITS OF THE USE OF PP	16
2.7 CONCLUSION	22

CHAPTER THREE: THEORETICAL AND CONCEPTUAL FRAMEWORKS	23
3.1 INTRODUCTION	23
3.2 CONSTRUCTING KNOWLEDGE WITHIN AN INTERPRETIVE PARADIGM	23
3.3 FRAMES OF ENQUIRY	24
3.4 THRESHOLD CONCEPTS	24
3.5 GROUNDED THEORY	29
3.6 CONCLUSION	33
CHAPTER FOUR: RESEARCH METHODOLOGY	34
4.1 INTRODUCTION	34
4.2 RESEARCH CONTEXT	34
4.3 RESEARCH METHODOLOGIES	35
4.4 TIMEFRAMES FOR DATA COLLECTION	36
4.5 OBSERVATION	37
4.6 QUESTIONNAIRES	37
4.7 INTERVIEW	40
4.9 ETHICAL CONSIDERATIONS	43
4.10 LIMITATIONS OF THE STUDY	44
4.11 CONCLUSION	44
CHAPTER FIVE: DATA ANALYSIS AND INTERPRETATION	45
5.1 INTRODUCTION	45
5.2 APPLYING GROUNDED THEORY	46
5.3 APPLYING THRESHOLD CONCEPTS	48
5.4 ANALYSING DATA ACCORDING TO THE NINE CONSIDERATIONS OF THRESHOLD CONCEPTS	50
5.5 THEMATIC ANALYSIS	55
5.6 CONCLUSION	62
CHAPTER SIX: SUMMARY, RECOMMENDATIONS AND CONCLUSION	66
6.1 INTRODUCTION	66
6.2 SUMMARY	66

6.3 RECOMMENDATIONS FOR TEACHING AND LEARNING	68
6.4 CONCLUSION	69
REFERENCES	71
APPENDIX A: PRE-TEST QUESTIONNAIRE.....	79
APPENDIX B: POST TEST QUESTIONNAIRE.....	94
APPENDIX C: QUESTIONNAIRE FOR EDUCATORS	98
APPENDIX D: OBSERVATION SCHEDULE	99
APPENDIX E: LETTER OF PERMISSION.....	100
APPENDIX F: INTERVIEW SCHEDULE	110
APPENDIX G: LETTER TO PRINCIPAL	112
APPENDIX H: ETHICAL CLEARANCE LETTER DOE (KZN).....	114
APPENDIX I: ETHICAL CLEARANCE - DUT	115
APPENDIX J: ETHICAL CLEARANCE – UKZN	116

LIST OF TABLES

Table 4.1 Data collection	36
Table 5.1: Features of grounded theory	47
Table 5.2: Criteria of grounded theory, in relation to this study	48
Table 5.3: Responses to working with a partner	55
Table 5.4: Self-rating: pre-test respondents	56
Table 5.5: Self-rating: post-test respondents.....	56
Table 5.6: Advantages of PP	57
Table 5.7: Popular reasons for working PP.....	60
Table 5.8: Paired partnership enjoy ability	61
Table 5.9: Popular reasons for enjoying PP	61

LIST OF FIGURES

Figure 3.1: An example of selected text with associated new code.....	32
Figure 5.1: Preferences of respondents	59
Figure 5.2: Post-test responses – likeability factor	60

GLOSSARY OF TERMS

In the context of this study, broad meaning that I have attached to the core concepts is as follows:

Educator	Includes a teacher at a high school and a lecturer at a tertiary institution.
Learners	Includes learners from high school and first year tertiary students.
Learning Environment	Includes both a high school class room and a university lecture theatre.
Peer-to-peer	learners studying programming in the same classroom/grade/programming course.
Grounded Theory	Using guidelines for the organization of data, theory is developed that provides relevant interpretations, applications, predictions and explanations.
Object-oriented programming	A computer programming paradigm based on the object-oriented approach, whereby objects have the responsibility of carrying out specific operations to solve a problem.
Object	Contains both data and operations in the same entity.

LIST OF ABBREVIATIONS

IT	Information technology
OO	Object Oriented
OOP	Object-oriented programming (concepts of classes, objects, events, inheritance, encapsulation)
PPA	Procedural Programming Approach (Concepts of sequence, selection and iteration)
POP	Procedural Oriented Programming includes PPA
PP	pair programming
CAP	collaborative adversarial pair

ABSTRACT

IT teaching strategies are unable to keep abreast of the ever-changing programming paradigms, programming languages and versions of software suites that are often “technologically hardware dependant” and costly to implement. Faced with ever increasing class sizes, lecture workloads and diminishing monetary resources coupled with reducing throughput rates in programming courses; IT educators are faced with challenges when teaching programming. The issue here maybe to move away from teacher centred learning to student centred learning. Pair Programming offers educators an opportunity to further enhance student centred learning. This study conducted an empirical study of “pair programming” in the teaching and learning of an introductory programming course in computer science with input from educators and learners. The purpose was to determine how a cooperative learning model can be used as a pedagogic tool for effective teaching and learning in a programming course. The study attempted to determine the impact of collaborative pair programming on students and whether IT educators can use pair programming as a teaching strategy. There was a pre-test for students to secure data and on how students attempted programming tasks. Thereafter pair-programming was implemented and a post-test was administered to determine the effectiveness of the intervention strategy. The research findings indicated that the educators and learners had a positive attitude towards the use of pair programming to support teaching and learning.

ACKNOWLEDGEMENTS

I am deeply indebted to the following people who inspired and helped me throughout this research:

- The participants, for their valuable contribution to this study.
- Dr D.W. Govender, my supervisor, for his guidance and support throughout this study.
- I wish to thank the University of KwaZulu-Natal (Faculty of Education-Edgewood campus).
- Department of Education(KZN)
- I also wish to thank SANPAD for funding which enabled me to do the necessary field work for this study and to fund my conference presentations.
- My wife, Vaneshree Govender – thank you for your love, patience and understanding;
- Klareesa, Shivar Govender, for being my inspiration; and
- My sister, mother and late dad, for their unconditional love and support.
- Finally, I thank God for granting me the strength and courage to persevere in completing this thesis.

Dedicated To
Vanesh, Klareesa, Shivar
&
Dr DW Govender

CHAPTER ONE

INTRODUCTION

1.1 INTRODUCTION

Among various aspects of information technology (IT) education, the ultimate outcome is to ensure good pedagogical quality and gainful employment by the IT student in the software industry. Every day there are an infinite number of ideas and software projects that industry and academia can conjure up; however, there must be an association between what academia conjures up and what happens in the software industry. Software projects researched by Bryant, Romero, and du Boulay (2006) encouraged PP amongst software developers. Laurie Williams, Kessler, Cunningham, and Jeffries (2000) validated anecdotal and qualitative assumptions that pair programming (PP) software products can be produced in less time, with higher quality.

As conscientious IT educators who require students to program individually during the teaching and learning process, should we not take our cue from the software industry which requires programmers to program in pairs/teams, and hence boost our IT learning outcomes and pass rates?

Over the years debates have persisted over the correct approach to teach Computer Science programming and the programming languages that should be used. According to Roy and Haridi (2003), the most popular approach to teach programming is in a single paradigm embodied in a single language. Cooper, Dann, and Pausch (2003) advocate an objects-first approach, while Howe, Thornton, and Weide (2004) consider the object-oriented (OO) and the component-first approaches to be most influential. Govender (2006, page 4) indicated in her research that there “exists tension between procedural paradigm and OO paradigm”. Ismail, Ngah, and Umar (2010), however, maintain that an OO approach to programming is not a good starting-point for introducing students to the basic concepts of programming.

While the debate rages on as to the challenges in higher post-school education and the insufficient levels of resources, examination results continue to diminish and dropout rates among first-year students increase (South Africa, 2012)

One strategy that has been found to be an effective means to combating the high failure rate and reducing the dropout rate – among other benefits to IT students – is a cooperative or collaborative

learning model. This model involves two or more individuals taking turns to help one another learn information (Horn, Collier, Oxford, Bond Jr, & Dansereau, 1998).

1.2 PURPOSE OF THE STUDY

The purpose of this study was to determine how a cooperative learning model could be used as a pedagogic tool for effective teaching and learning of an introductory programming course at secondary schools and institutions of higher learning within KwaZulu-Natal. The study aimed to determine the impact of collaborative PP on students, and whether IT educators could use PP as a teaching strategy. Further, this study examined the perceptions and experiences of IT educators and learners in KwaZulu-Natal who used PP as a tool to facilitate the teaching and learning processes with respect to the OO paradigm.

1.3 PROBLEM STATEMENT

At a university of technology's IT department (Campus A), Microsoft C# and Java were adopted as the programming languages in introductory programming courses. At this campus computer problems were solved with a particular programming language and within a particular programming paradigm. The lecturers assign problems to students in a programming language course, and students then independently solve or attempt to solve the problem and code the solution in the particular programming language.

Students in schools and tertiary education program on their own, but software houses require individuals to program as part of a team. This disjuncture in between how they are required to work in industry and how students are trained. A possible consequence maybe that PP will improve the pass rates.

1.4 RATIONALE FOR THE STUDY

As a researcher and lecturer at a university of technology that is experiencing a dropout rate of almost 50% at first-year level, and where the first-year teaching pedagogy remained focused on the solitary/independent/individual programming approach, the researcher was convinced that unless there were to be a pedagogic intervention in the teaching of programming the dropout rate would not change. Mentz, van der Walt, and Goosen (2008) observed a strong tendency among teachers to

depend on strategies which focused on the individual when teaching programming skills — they are generally inclined to favour individual problem solving and individual practice of programming skills.

The rationale for this study was that students perceived programming to be difficult (Koorse, Calitz, & Cilliers, 2010) coupled with the increased dropout rate among programming students and decreased throughput rate of programming qualifications (MacGregor, 2007a).

The University of South Africa (UNISA) is a distance, correspondence university and in 2007 it announced that it would spend nearly R50 million in intervention strategies to support its programmes (MacGregor, 2007a). Peer-to-peer learning was one intervention strategy conducted in an informal setting at regional learning centres, aimed at improving the pass rate and reducing the student dropout rate.

Kinnunen and Malmi (2006) indicated that many institutions report dropout rates of 20–40% or even higher in introductory programming courses. Furthermore, there is overwhelming evidence (Cooper et al., 2003; Govender 2006; Havenga & Mentz, 2009; Kolling & Rosenberg, 2001; Okur 2006) that supported the fact that both students and educators considered programming difficult to learn and teach, and that an intervention strategy was required to improve students' pass rates. This bears testimony to the fact that educators and academics are seriously concerned about the high student dropout rate and dwindling numbers of new students enrolling for Computer Science programming courses. The declining graduation numbers in the Computer Science discipline has also been documented by several authors (Howles, 2009; McKinney & Denton 2004; Ventura & Ramamurthy 2004)

At Campus A earlier programming courses were consistent with command-driven and procedural programming language. In response to the requirements of the software industry and the low pass rate, the first-year IT curriculum was amended to include a course in OO programming. However, students remained unable to meet the requirements of the programming course.

While searching for and narrowing a field of research for this study, the researcher was introduced to the concept of PP as a strategy with the potential to improve the situation. This was in 2010 at a university where PP was being researched as a teaching strategy to be implemented to teach introductory IT students. Regionally, that PP had not been used in introductory IT programming courses, and this formed the rationale for the study.

1.5 OBJECTIVE OF THE STUDY

The objective of the study was to examine the use of PP to support teaching and learning of an introductory programming course. PP has been demonstrated to be beneficial (Mentz et al., 2008; Nosek 1998; Laurie Williams et al., 2000). The major advantages of PP were higher quality of programs, decreased time to complete programs, improved understanding of the programming process, improved course completion rates and improved performance on exams (Preston 2005). Mentz et al. (2008) showed that PP was successfully implemented in teacher training in North West Province, South Africa. Breed et al. (2013) research indicated that meta-cognitive skills during PP can result in increased productivity. This positive feedback surrounding the use of PP was motivation for this study.

The objective was to provide learners and educators at schools and tertiary institutions with a strategy that they could adopt when teaching and learning introductory programming courses.

1.6 THEORETICAL AND CONCEPTUAL FRAMEWORK

The theoretical and conceptual framework for this study comprised threshold concepts in Computer Science, and grounded theory.

Threshold concepts are those which might be used to organise the educational process, and are likely to be transformative, integrative, irreversible, and potentially troublesome for students, and are often boundary markers (Cousin & Meyer, 2006). These attributes are discussed in detail in Chapter Three.

Grounded theory was used to develop theory that provided relevant interpretations, applications, predictions and explanations (Meyer & Land, 2005). Data are analysed and discussed according to these criteria in Chapter Three.

The interpretive paradigm was chosen for this study because it was concerned with descriptions that produced deep understanding and emphasised interpretation of data from educators and learners.

It is within the boundaries of these theoretical frameworks that the research questions were answered.

1.7 KEY RESEARCH QUESTIONS

In this study key research questions were as follows:

1. What are learners' experiences of solving programming tasks?
2. What teaching strategies are being used to teach OO programming?
3. How does PP enhance problem solving in OO programming?

1.8 RESEARCH DESIGN AND METHODOLOGY

1.8.1 Research methodology

This study used a mixed-methods approach, where both qualitative and quantitative methods were used to address the research questions.

Duffy (1987) summarised the relationship between quantitative and qualitative research: “quantitative research is used to evaluate objective data consisting of numbers, while qualitative research deals with subjective data that are produced by the minds of respondents.”

Qualitative research methods involve collecting textual or verbal data and observation of people followed by careful description and analysis(Boeree, 2008). For this study the data from the interviews and observations were analysed using qualitative research methods. Quantitative methods were employed in analysis of data collected from the questionnaires.

1.8.2 Context and sampling

The participants were Grade 11 and Grade 12 IT learners from secondary schools, and IT students from a university of technology and from a university.

1.8.3 Methods of data collection

In order to obtain data that would further the aims and strengthen this study, it was decided that it was necessary to use more than one data collection method. The study used interviews, questionnaires and observations. The researcher observed the learners programming in pairs, and learners complete the pre-test and post-test questionnaires. Educators were interviewed before and after the intervention.

The observations, interviews and questionnaires enabled the researcher to provide answers to the research questions.

1.8.4 Data analysis and interpretation

Data analysis involved organising, analysing and interpreting data (McMillan & Schumacher, 1993). Data from the questionnaires and observation sheets were captured on a spreadsheet as numerical data, which facilitated statistical representation in percentages and graphs. The data from the interviews were transcribed, and once the transcripts were completed the researcher looked for themes and categories that were associated with the theoretical framework, guided by the research questions.

1.8.5 Ensuring the trustworthiness of the study

Application of a multi-method approach allowed for a comparison of data – referred to as triangulation Krefting (1991). According to Denzin and Lincoln (2000) triangulation is a means of ensuring concurrent validity and prevents personal bias. Validity refers to the appropriateness of the conclusions claimed from the analysis of the collected data Jandaghi and Shaterian (2008). This has to do with whether the research methods, approaches and techniques used were appropriate to the study conducted. The researcher interviewed educators and observed learners. The interviews were then transcribed. Data was collected from observation of lessons and from the questionnaires. The reason for this was that the researcher was able to acquire the same data using different research instruments and different forms, thus enabling the research to acquire triangulation, which enhanced the validity of the data collected Klopper (2008). Furthermore, the data sample included two tertiary institutions and three secondary schools, which further enhanced the reliability of this study. The collection of data from differences sources added to the validity, reliability and trustworthiness of the study.

1.8.6 Ethical issues

This study carried the approval of the Department of Education to conduct this research with learners as participants (Appendix H). Each learner participated willingly in the study and both parents and learners completed a consent form (Appendix E). The consent form also guaranteed confidentiality of participants. This study also carried the approval of the Ethical Committees and Research Directorates of the higher education institutions where the research was conducted (Appendix J)

1.9 STRUCTURE OF THE STUDY

This study comprises six chapters, which are broadly outlined below.

Chapter One introduces the study and presents the research topic, research questions, problem statement and rationale for the study.

Chapter Two provides a literature review of the research topic, including collaborative and cooperative learning, and PP and its increased use in academia and the software industry. This literature review also reveals results of some previously conducted studies on PP in academia and the software industry.

Chapter Three discusses the framework for the investigation, knowledge and skills in cooperative learning strategies, including threshold concepts and grounded theory.

Chapter Four sets out the research methods used to explore the research questions and give an overview of the research design; it also deals with the interpretative paradigm, data collection and data analysis methods.

Chapter Five presents an analysis of the results of implementing PP and evaluation and analysis of the research questions. Tables and graphs are used to show the findings from the questionnaire, and the thematic analysis is applied to data from the interviews. This is followed by a discussion of the research findings according to the principles of grounded theory and threshold concepts, and with reference to the literature review.

Chapter Six provides an argument for the use of PP in classrooms and lecture rooms to support teaching and learning. Some recommendations are made for further study and limitations are discussed.

CHAPTER TWO

LITERATURE REVIEW

2.1. INTRODUCTION

This section consolidates ideas emerging from developments that have so far been made around the phenomenon of PP in various fields from a theoretical perspective. Drawing from a variety of studies conducted by numerous researchers, the purpose of this section is to attempt to establish the nature of PP and the variables that influence this mode of learning. This chapter seeks to unpack theoretical assumptions developed around PP from the perspective of researchers who have conducted in-depth studies that explore the working of this phenomenon as practised in diverse contexts of learning.

Literature reviewed in this chapter provided a comprehensive view of the key arguments and findings of a number of researchers that were used to generate assumptions necessary to inform the outcomes of this study. The literature informed the structure taken by this study as it draws on how other researchers whose work was used to develop this literature conducted their studies from both the theoretical and methodological perspectives.

Programming is a challenging task and programming courses are generally considered problematic by many learners (Govender, 2006; Macgregor, 2007b; Havenga & Mentz, 2009). This research study therefore explored PP through students' and teachers' experiences not only as an alternative to traditional methods to teaching and learning programming, but also as a way of complementing traditional methods of teaching and learning programming. In this study almost 40% of the respondents from the pre-test rated themselves as struggling with computer programming.

Cooperative learning, and more importantly PP, has opened up new possibilities for learning programming. Collaborative learning lies at the centre of human development, motivated by the desire for education that seeks to construct knowledge for the learning society (Education, 2003). Learning and the construction of knowledge have of late brought about new concepts, for example the introduction of mobile online learning, which in turn have given rise to changes in educational objectives.

However, the researcher must emphasise that for PP to deliver desirable learning outcomes, practitioners need to develop a pedagogic will and passion to motivate students to participate in this mode of learning, while institutions help level the ground by establishing resourceful contexts and customised practitioner training in the PP paradigm. In this study educator support has been

considered (Appendix D: Interview schedule with educators), and three of the five educators indicated that they received no support from teachers, management or surrounding schools. This study therefore played an important role in ‘practitioner training’ with respect to pair-programming.

The effective use of PP in support of classroom teaching by integrating technology with an appropriately considered pedagogical approach was central to teaching and learning programming in this study. Literature reviewed in this chapter sought to explore how PP made way for extended learning opportunities. Improving the quality of education is a broad policy objective in South Africa (Education, 2003).

This chapter together with Chapter Three was structured in such a way that literature surveyed gave consideration to relevant aspects relating to PP, collaborative learning, students’ experiences of learning programming in higher education, and gaps in the literature, as well as a framework for researching PP. This broad framework has been outlined, describing the elements and dimensions that influenced student learning when learning was conducted in a paired-programming environment.

This study enquired into students’ and educators’ experiences of PP at secondary and higher education institutions, and drew on the research conducted by Mentz et al. (2008). Pair programming was successfully implemented in teacher training in North West province, and research proved that it improved learners’ understanding of programming principles (Mentz et al., 2008). Mentz and Goosen (2007) indicated that IT teachers in Gauteng and North West did not have the knowledge and skills to apply cooperative teaching strategies, and that teachers were unaware of the value of applying pair-programming in their classes. In addition, anecdotal evidence suggested that when an educator ‘debugs’ a student’s syntactical, logical and runtime errors, the student and educator are engaged in collaborative PP. Hence, although some of the educators from the sample population were unaware of PP and cooperative programming principles, it was noted during the data collection the educators were applying the principles in an informal/intuitive strategy. When asked whether PP assisted in assisting slow learners, an educator from the sample population stated:

“Yes, it pushes them to move faster and allows me to teach other pairs”.

This study aimed to formalise and create greater awareness among educators with respect to the benefits of PP. The educators’ responses are further elaborated upon in Chapter Five.

2.2 DIFFERENCE BETWEEN COLLABORATIVE LEARNING, COOPERATIVE LEARNING, PP AND CAP PROGRAMMING

The collaborative learning literature states that student interaction occurs in small teams of four and occasionally five members, and that students often practice in pairs within the teams. Davidson (1994). A review of a century of research on collaborative learning (from the late 1800s to the late 1900s) led to the observation that working together to achieve a common goal produces much higher achievement and greater productivity than working alone (Johnson & Johnson, 1992).

Research supports the premise that collaboration is an effective pedagogy for introductory programming (McDowell, Werner, Bullock, & Fernald, 2002;Cliburn, 2003; DeClue, 2003). The collaborative learning research literature identifies cooperative behaviour as students discussing problems together and correcting any misconceptions or mistakes, and it has been identified as a one of the five critical attributes common to successful collaborative learning (Davidson, 1994).

Vandegrift (2004) states that “PP is a form of collaborative learning in which groups consisting of only two members – a driver and a navigator – work together on the same computer to complete the same project”. Hanks and McDowell D(2004) further elaborated that “Each member also has individual responsibilities and roles to perform”. Preston (2005) observed students who worked in pairs, and who provided feedback did not appear to be common practice in PP; this author therefore concluded that feedback should be included in cooperative behaviour and performance of the roles of navigator and driver. Pair programming is an application of collaborative learning (Preston, 2005).

Laurie Williams, Wiebe, Yang, Ferzli, and Miller (2002) however, loosely define PP and collaborative programming interchangeably: “Pair or collaborative programming is where two programmers develop software side by side at one computer”.

Swamidurai and Umphress (2012) propose an amendment to PP called collaborative adversarial PP (CAP). This is included in the discussion with respect to the definition of this study because it highlights a significant enhancement to the definition of PP and further redefines the scope of PP.

The differences between PP and CAP are as follows:

- In PP the pairs collaborate in all phases of software development (analysis, design, implementation, testing), whereas in CAP pairs start together in the design phase, then split into independent roles, then join again for testing.

- PP test-driven development follows a cycle of test, code, and refactor, whereas in CAP the developer self-inspects. Refactoring is the process of changing software's internal structure in order to improve design and readability and reduce bugs, without changing its observable behaviour. It helps developers to review someone else's code (Fowler, 1999) and hence CAP has an additional refactoring phase.
- In PP a new development cycle begins again after the test phase, but in CAP a post-mortem phase begins (review of source code and test cases).

2.3 HISTORY OF PP

There has been a decade of research into PP and its usefulness and effectiveness in both academic and industrial settings (Hannay, Dybå, Arisholm, & Sjøberg, 2009 ;Salleh, Mendes, Grundy, & Burch 2010). Beck (1999), however, claims that the history of PP stretches back to punch cards. Researchers at Microsoft Begel and Nagappan (2008) found that most research focused on academic environments, and there were limited studies about PP in industry. This study has focused on academia, i.e. secondary and tertiary institutions of learning; however, academics must always bear in mind that the ultimate client of programming is the software industry.

As far back as 1978 there was a conference on programming languages. The Conference on History of Programming Languages Wexelblat (1978) described the 13 computer programming languages present at the time. This study does not focus on programming languages, but rather on a strategy that a teacher can adopt to teach a particular programming language.

Although the history of PP stretches back to punch cards, it emerged as a viable approach to software development in the early 1990s when it was noted by Williams et al (2000) “as one of the 12 key practices promoted by extreme programming”. Swamidurai and Umphress (2012) claim that PP “has been widely accepted as an alternative to traditional individual programming”.

2.4 SHORTCOMINGS OF PP

To negate the shortcomings of PP and make it more effective in the teaching and learning of programming, PP should be incorporated into the principles of cooperative learning. This conclusion was reached by Mentz et al. (2006). Hence, before discussing the benefits of PP this research will attempt to summarise the shortcomings of PP and the rationale for cooperative learning.

A potential shortcoming as described by Mendes , Al-Fakhri , and Luxton-Reilly (2005) is the “danger of the academically stronger member of the pair doing all or most of the work”. Laurie Williams et al. (2002) included the following in the discussion: “The possibility of a student remaining in the same role throughout and therefore failing to learn the skills associated with the other role, students reversing roles without approval of the instructor.” This would lead to inadequate PP and the concern is addressed in chapter six. There is also some evidence that PP lead to a reduction in the effort of at least one of the programmers in a pair (Hannay et al, 2009). Williams et al. (2002), has stated that students preferred to work on their own, either because “intra-pair pressure prevents them from cheating or that their personal achievement will be better”. Vandergrift (2004) simply stated that a student would work on his own because he did not have the time to setup a meeting with a partner. This research did not encounter such shortcomings because the educators had complete control over the structuring of PP sessions.

Stevens and Rosenberg (2003) describe “Go make a cup of tea syndrome: the more experienced programmer does all the work while the novice programmer falls behind”. They also discuss clashing coding styles and an unenthusiastic PP partner. However, these did not emerge in this study and it may be assumed that the non-competitive academic environment contributed to this.

Randall (2003) found “the worst pairing is of two inexperienced programmers that are near the same skill level”. These PP partners will lack skills and will require outside intervention. Conversely experienced programmers may not program in pairs due to conflict in their egos.

Begel and Nagappan (2008) quite succinctly describe PP partners as “he attributes of a good programming partner as similar to those of a good spouse”. They further to describe problems perceived with pair programmers as “individual anxiety at working closely with someone else and organisational anxiety over allocation of funds to PP”. The problems mentioned by these Microsoft researchers were also cited by Mentz and Goosen (2007). The associated problems of PP formed the framework of the data collection instruments in Chapter Four.

In order of occurrence, Begel and Nagappan (2008) reported the following problems associated with PP:

- “It is hard to adopt PP;
- There may be limited office space;
- It is hard to think with another person working over your shoulder,
- It requires a commitment from the entire team,
- Partners can be overcritical,

- It is difficult to get management to agree to let you program in pairs, and you cannot easily do independent work while pairing,
- Some people are afraid of PP,
- Programmers feel a lower sense of ownership over the code:
- Pairing suffers when there are too many changes in the product cycle,
- Partners may end up competing with each other,
- Programmers feel that people should sometimes work alone,
- They worry about an unequal commitment level in the partner,
- Long-term pair compatibility,
- Having a passive partner, Lack of privacy,
- Hard to pair when the team has an odd number of people,
- There is less work on upfront design, and hard bugs are not found,
- It is stressful,
- Hard to operate when a partner is absent,
- People do not know about PP,
- It can be difficult to share driving duties (they cannot multi-task),
- They may need more than two people when working on a problem,
- They may need to rework some code written by the partner,
- It is too easy to stop pairing,
- It requires discipline,
- Sometimes it does not work if the partner is not compatible, and
- It can deliver fewer features”.

Swamidurai and Umphress (2012) point to three significant disadvantages of PP

- Two developers must be at the same place at the same time, which is often not possible.
- It requires an enlightened management that believes that letting two people work on the same task will result in better software, a significant obstacle since software is measured by tangible benefits.
- Empirical evidence is mixed: some research supports costs and benefits while other research shows no statistical difference between pair and solo programming (McDowell et al., 2002; Nosek 1998; Williams et al., 2000)

This study showed a distinct difference between pair and solitary programming, highlighted in the scoring patterns of the group-rated exercises in Chapter Five.

Whilst Swamidurai and Umphress (2012) may point to disadvantages of PP, they see CAP as an alternative to traditional PP where PP is not beneficial or not possible to practice. Furthermore, CAP has similarities to PP.

Solutions have been offered for the disadvantages of PP and related problems. Williams et al.(2002) suggested, “that students should be required to report on their own contributions to the project as well as on those of their partners”. It was recommended a strong supervisory role for the instructor/lecturer/facilitator, and also for the purpose of ensuring that the members of a pair took turns at being either driver or navigator. Howard (2006) contended that “individual assessment should serve as the ultimate source of each member’s personal accountability”. “Peer evaluation can be invaluable in this process” (Cliburn, 2003). Mentz , Van der Walt , and Goosen (2008) concluded that “...these solutions seem to form part and parcel of cooperative learning”. A discussion of the five principles of cooperative learning now ensues.

2.5 FIVE PRINCIPLES OF COOPERATIVE LEARNING

2.5.1 Positive interdependence

Both members of a pair, the driver and navigator, should understand that one of them cannot succeed unless they both do (David W Johnson , Johnson & Smith 2007) . The facilitator should ensure that the members take turns at being driver and navigator (Howard, 2006). Goal setting, incentives or bonuses for excellent work, collective responsibility, shared resources, mutual support and encouragement are required for the achievement of success (Veenman, Van Benthum, Bootsma ,Van Dieren & Van der Kemp 2002). Both partners should understand that their work will be assessed from time to time, and they should be given insight into how the assessment will work.

2.5.2 Individual accountability

Measures should be in place to ensure that both partners participate equitably (Davis, 2009) and contribute towards achieving the expected outcomes. Performance assessment of each partner’s work will ensure that both will contribute to the effort. The results should be given to both partners to reflect upon (Johnson et al, 2007). By doing this the facilitator ensures that the partners keep each other accountable. Accountability can be reinforced by the facilitator requesting either partner to demonstrate and explain their program to the rest of the class, and this can be followed by peer assessment. Bonus points given for excellent reporting can serve as a further incentive and promote awareness of responsibility (David W Johnson et al., 2007). For individual assessment each member

of a pair can also be required to write a program similar to that written by the pair. A student's personal accountability can be enhanced if the results of each member's individual assessment is given to the pair to reflect upon. If all these measures are in place, competition as well as blaming the other can be avoided (David W Johnson et al., 2007) .

2.5.3 Face-to-face interaction

The partners, physically, work together, and encourage and facilitate each other's efforts to achieve success (David W Johnson et al., 2007). This results in providing assistance to the partner, exchanging resources, and challenging conclusions, reasoning and strategies. They should also be guided to understand how to act in the roles of driver and navigator respectively, how to assume responsibility, how to share roles, set targets for themselves as individuals and as a pair, divide tasks between them, and communicate with each other (McWhaw, Schnackenberg, Sclate , & Abrami 2003). This in turn will increase willingness to ask for assistance, and lead to an increase in confidence to suggest ways of improving a programming code and of solving problems. It will also result in reduced stress. To achieve all of this, each partner should understand exactly what is expected of them as individuals and as a pair, and they should master the skills required to achieve the objectives of the project. To facilitate this the lecturer should provide time-slots during which PP can take place and be observed by her, such as during scheduled contact sessions or laboratory sessions (Preston 2006).

2.5.4 Development of good social skills

The partners should communicate clearly and regularly with each other; they should develop interpersonal skills and learn to trust each other and to resolve conflict amicably (David W Johnson et al., 2007). They should learn to accept criticism graciously and to criticise in an acceptable manner, to test ideas (Williams & Kessler 2003), formulate a problem and discover appropriate strategies for overcoming obstacles, and to encourage and compliment the partner on work well done. To ensure the optimal development of these skills, the partners should often reverse the roles of driver and navigator (Williams & Kessler 2003). As their social skills improve, so the enjoyment factor increases and the partners become more motivated for the task. This study also found a surprisingly high percentage of respondents (86.8%) who stated that they enjoyed programming with a partner.

2.5.5 Group processing

The partners should periodically reflect on how well they are functioning and how they plan to improve their achievements (D W Johnson & Johnson 2009) . The facilitator should give them time for such reflection. During PP activities, the facilitator should also be constantly available for consultation, guidance and assessment.

2.6 BENEFITS OF THE USE OF PP

In the past popular paradigms included imperative programming, functional programming, logic programming and concurrent imperative programming (Van Roy & Haridi 2002). Empowering learners to become self-sufficient has always been and will always be a primary objective of education, and PP emerges among a number of recent innovations in the field of teaching and learning programming that have the potential to assist in this.

Pair programming is often used in professional software development communities and appears in commercial training environments as well as in some undergraduate and high school classrooms (Nagappan et al., 2003). Chong and Hurlbutt (2007) noted that PP is a common professional practice in the workplace. When employees programme in isolation they might not experience collaborative solving of problems, communicating their understanding verbally, and resolving disagreements with their peers, all vital skills in the world of work. If one has to correctly assume that one of the primary aims of an education is gainful employment, then introducing PP at institutions of learning is preparation for the world of work. Williams et al. (2000) stated that among the benefits of PP are increased self-confidence and an interest in IT. Breed et al. (2013) research indicated that meta-cognitive skills used during PP can result in increased knowledge productivity.

Ismail, Azilah, Naufal, and Kelantan (2010) state that the main cause of difficulty in understanding programming and coding is the ‘inactive involvement’ of students during programming tutorials. Nosek (1998) found that PP outperformed individual programmers and that code from individual programmers had more errors than code programmed by PP. A similar result was found in this study, as Chapter Five will reveal. Braught, Eby, and Wahls (2008) also found that students with low standardised test scores in Mathematics showed significant improvements in individual programming skill when enrolled in classes that use PP and it was also found that drop-out rates decreased and academic achievement was enhanced. Similarly, Nagappan et al. (2003) showed that a benefit of PP was improved course completion rates. Furthermore, McDowell et al.(2002) noted that improving

student experiences in introductory classes also increased the likelihood that students will continue taking more advanced classes in computer programming.

An interesting (and gender-based) point of note was increased efficiency in helping female students to work in programming tasks (Berenson, Slaten, Williams & Ho, 2004). Female students working in pairs enjoy the programming process (Werner , Hanks , & McDowell 2004); the enjoyment comes from the usefulness of the program and teamwork (Ho, Raha, Gehringer, & Williams, 2004) . Carter (2006) found a larger proportion of women than men reject Computer Science because they prefer a more ‘people-oriented’ field of study. Liebenberg, Mentz, and Breed (2012) also concluded that PP contributed positively to school girls’ enjoyment of programming and of the subject of IT. Female students observed that they were more productive when working collaboratively, taking less time and producing a higher-quality product. With increased productivity, these women experienced more confidence and consequently a greater interest in IT careers (Berenson et al., 2004). When exposed to PP women reported greater confidence and demonstrated a greater likelihood to take up Computer Science in future (Braught et al., 2008). Considering that in the United States of America women make up less than one-fifth of undergraduate Computer Science classrooms (Zaidman 2011), and the fact that PP is a more ‘people-oriented’ strategy, then employing techniques of PP augers well for the inclusion of more women in undergraduate courses and later in the workplace. In this study women made up approximately one-third of the sample population. IT appears to be very male dominated; this is further elaborated upon in Chapter Five.

2.6.1 Improved understanding of programming principles

Research supports the assertion that PP can improve learners’ understanding of programming principles. The research of Barker, Garvin-Doxas, and Roberts (2005) indicated that allowing students to work PP environments or groups, and allowing them to articulate what they learnt, improved their understanding of programming principles. Tillema and van der Westhuizen (2006) found a similar benefit: students working in a study group and not in isolation had a greater awareness of and sensitivity towards a problem. Preston (2006) also indicated that PP has been effective for teaching students to program and understand basic programming syntax and principles.

McDowell et al. (2002) concluded that scores for PP sections were significantly higher than those from non-pairing classes, and that the process of working collaboratively improved the quality of programs. It was also found that the drop-out rates decreased and academic achievement was enhanced. Nagappan et al. (2003) cited improved course completion rates as a benefit of PP. Bain

(2004) stated that “undergraduates learn most effectively when they can work collaboratively with other learners to grapple with problems”. This study found a similar result.

Berglund, Box, Eckerdal, Lister, and Pears (2008) are of the opinion that educators should encourage learners to engage in their own learning. Kinnunen and Malmi (2006) identified students’ social interactions as a way of decreasing drop-out rates. PP provides this opportunity for students to not only engage and interact in their own learning but also to engage with their learning in the company of a friend/peer or colleague. Programming does not become a solitary experience and the educator does not adopt the teacher/educator-centred strategy. An important point of particular note with respect to this study in KwaZulu-Natal was the fact that the researcher together with educators observed students in pairs conversing in their mother-tongue language, either isiZulu or isiXhosa, when the pair discussed a particular obstacle in their programming task. This social interaction complemented the students’ ‘cultural interaction’ of language. Furthermore, the literature surveyed in this chapter neglected to determine the effects of mother-tongue instruction among pairs; this is further elaborated upon in Chapter Five and Chapter Seven.

Preston (2005) cited the following benefits to the use of PP:

- “Higher quality of programs;
- Decreased time to complete programs;
- Improved understanding of the programming process;
- Improved course completion rates; and
- Improved performance on exams”.

Mentz et al. (2008) concluded that “the experimental group not only outperformed the control group but there were also fewer dropouts”.

Cockburn and Williams (2001) summarized the significant benefits of PP as follows:

- Many mistakes get caught as they are being typed in, rather than in the quality assurance test or in the field (continuous code reviews);
- The end defect content is statistically lower (continuous code reviews);
- The designs are better and code length is shorter (ongoing brainstorming and pair relaying);
- The team solves problems faster (pair relaying);
- The people learn significantly more about the system and about software development (line-of-sight learning); and
- The project ends up with multiple people understanding each piece of the system.

2.6.2 Benefit to inexperienced and experienced programmers

Padmanabhuni, Tadiparthi, Yanamadala, and Madina (2012) conducted research with pair programmers across different student levels. They concluded that PP is an efficient technique in programming and that junior students can gain more knowledge from senior students; they also learn much better than by reading books.

Furthermore, experienced programmers have also been quoted as stating that they too benefit from pairing with junior programmers (Lang & Ottinger 2011). Even inexperienced novices contributed to an experienced programmer, according to interviews by Cockburn and Williams (2001).

While this study only studied students on an equal academic level, it is interesting to note that academics may wish to consider pairing programmers across different academic year groups. This, however, falls outside of the scope of this study.

2.6.3 Increased social development & enjoyment of programming

Pikkarainen, Haikara, Salo, Abrahamsson, and Still (2008) found an interesting benefit to PP, with observed sessions showing high verbal interactions. Pair programmers were shown to produce in excess of 250 verbal interactions per PP hour, and both partners collaborated on 93% of their tasks. Cockburn and Williams (2001) also found that programmers learn to work and talk more often when they are together, and produced better results. Furthermore, there was a higher satisfaction level among people working in pairs than among those working alone. Cockburn and Williams (2001) reported that PP partnerships were more confident and enjoyed programming PP more than when they programmed alone. Liebenberg (2010) indicated that PP results in greater enjoyment of programming. This study also concluded that when students enjoy programming, they tend to work more diligently and excel in the programming module, which is reflected in their marks.

Lang and Ottinger (2011) noted that “having a whole team in a room can be noisy and distracting, while a focused pair can more easily block out distractions than an individual and that people are also less likely to interrupt a pair deep in work and conversation than an individual sitting alone”. Pikkarainen et al. (2008) noted high verbal interaction. Lang and Ottinger (2011) noted that the conversation among paired programmers prevented disruptions from other people/programmers. This augers well for dispelling the myth that programming is a solitary activity performed by antisocial software engineers sitting and coding in lonely and quiet offices and grudgingly producing lines of code.

2.6.4 Costs related to software development and programming

Despite going to the extent of overtime and extra costs, approximately 30–40% of software projects are completed, and others fail (Molokken & Jorgensen, 2003). Cockburn and Williams (2001) noted that managers may view programmers as a scarce and a costly resource, and may not want to implement PP principles. However, their research concluded that “the development cost for the benefits of PP is not the 100% that might be expected, but is approximately 15% and that this is repaid through shorter and less expensive testing, quality assurance and field support”.

Galorath and Evans (2006) found, among other reasons for software project failures, insufficient requirements engineering, poor planning, sudden decisions and inaccurate estimations. Jorgensen & Shepperd (2007) found that inaccurate estimation is the root factor of failure in most of the software projects that do fail. This study has found that PP can reduce the time spent by student programmers on academic software projects.

Macgregor (2007a) reported that “40% of South African students drop out of tertiary institutions in their first year of study”. This study’s sample population also indicated a high dropout rate amongst its first year students. High student drop-out rates are a major problem and intervention strategies like PP would improve pass rates and simultaneously reduce costs and improve the throughput rate of tertiary institutions.

Kent Beck (1999) credits much of the success in extreme programming to the use of PP where programmers worked in teams programming, designing, or testing and after a designated time period partners reversed their roles. Programming code is collaboratively reviewed before it is integrated. Anderson, Beattie, and Beck (1998) exclusively and successfully implemented PP in the Chrysler Comprehensive Compensation system. Success in industry equates to increased profits and success at tertiary institutions equates to increased throughput rates.

As a point of note, it was the benefits of PP in industry that motivated academics (and this study) to apply the techniques of PP in higher educational institutions (Salleh et al., 2010). If PP can work in industry and we at higher educational institutions are preparing students to find employment in industry, should we not be adopting successful industry standards of work ethics and practices?

McDowell, Werner, Bullock, and Fernald (2002) research concluded that scores from PP sections were significantly higher than the scores from non-pairing classes, and that the process of working collaboratively improved the quality of programs. It was also found that drop-out rates decreased and

academic achievement was enhanced. Nagappan et al. (2003) stated that a benefit of PP was improved course completion rates. Bain (2004) stated that undergraduates learn most effectively when they can work collaboratively with other learners to grapple with problems.

Breed's et al. (2013) research indicated that meta-cognitive skills during PP can result in increased knowledge productivity. Increased productivity and decreased drop-out rates translate into budgetary cost savings for any organization, either academic or industrial. Employing the principles of PP can assist in cost savings by reducing drop-out rates and increasing the chances of software companies producing code and programs that are successfully implemented. This augers well for increased profits and decreased downtime in the software industry.

2.6.5 PP can assist academics with student consultation

Consultations with students in higher education environments featuring large class sizes is often crippled by time limitations, making it necessary for academic teachers to explore alternative ways of complementing traditional face-to-face consultations with students with computer-mediated models of communication (Field, 2005).

A study at the University of the Witwatersrand indicated that person-to-person consultation with students in overcrowded lecture halls is virtually out of the question. The ever-increasing student population and duplication of queries likely to be raised result in an extra load being exerted on academics to attend to each student (Thatcher 2007).

The University of South Africa (UNISA) is a distance, correspondence university, and in 2007 UNISA announced that it would spend nearly R50 million on intervention strategies to support its programs (Macgregor, 2007b). Peer-to-peer learning was an intervention strategy that was used informally at regional learning centres, aimed at improving the pass rate and reducing student drop-out.

Anecdotal and empirical evidence by Salleh (2008) showed that PP made significant contributions to performance in introductory programming courses in Computer Science. Similarly, preliminary qualitative evidence has suggested that not only can PP improve learning among introductory programming students, it also benefits teaching by reducing the amount of student consultation time. Students would inquire from the paired colleague before requesting lecturers' assistance.

Ramsden (1992) second principle for effective teaching in universities refers to concern and respect for students and student learning and the strategies employed were conversational framework, consultation and negotiation processes. This occurred between the educator and the student.

If one adjusted the above to incorporate PP instead of consultation and negotiations, one would now have a strategy that academics can employ to effectively support the teaching and learning of programming between students. This desire for effective support for programming students is one of the reasons that motivated the researcher to conduct this study. This support can best be achieved by considering students and student learning as important constituents of the pedagogic situation. Pair programming offers one such solution to complementing traditional lecturer-to student consultation and decreasing the drop-out rates of IT students.

2.7 CONCLUSION

The shortcomings and problems associated with PP suggest that further research is required to explore strategies to provide solutions and hence negate these. The aforementioned studies have shown that integration of PP makes a positive contribution to teaching and learning, because it provides an effective means of communication and full utilisation of resources by learners. The researcher's experience with implementing PP in tertiary education also suggested that this teaching strategy is important in teaching and learning.

Review of the literature revealed that PP has been used in the software industry and educational institutions to either generate new software projects and/or to support teaching and learning of programming concepts. Relatively few studies have investigated the use of PP as a strategy in teaching at secondary school level, and even fewer have been conducted in the South African setting. While the adoption rate of PP has been increasing rapidly, very little is known about how this teaching strategy benefits teaching and learning or how it brings about change to the current classroom/lecture room practice.

In this study the researcher investigated respondents' (learners' and educators') perceptions and experiences of using PP. The research intends to add to previous research on the use of PP in education, since to date students' and educators' experiences and the efficacy of PP have been subjected to limited research, especially in the KwaZulu-Natal region of South Africa.

CHAPTER THREE

THEORETICAL AND CONCEPTUAL FRAMEWORKS

3.1 INTRODUCTION

The purpose of this chapter is to describe the research design of this study. Research design refers to the method(s) and technique(s) used to collect, analyse and interpret data and a paradigm refers to how a researcher would view that which they study (Du Plooy, 2009). Different paradigms can be used to explain or predict phenomena and in this study it refers to the way in which the research was conducted.

3.2 CONSTRUCTING KNOWLEDGE WITHIN AN INTERPRETIVE PARADIGM

This study was constructivist-based within an interpretive paradigm. The ontology associated with the interpretive paradigm refers to the “internal reality of subjective experience” (Terre Blanche & Durrheim 2002). Furthermore, this paradigm focuses on the construction of knowledge in situations where problem solving is required, in this case programming tasks with specific reference to programming in pairs.

Both quantitative and qualitative research methods are suited to this paradigm. Babbie (2010) states that "Qualitative research is the collection, analysis, and interpretation of data by observing what people do and say and it's a generic approach in social research according to which research takes as its departure point the insider perspective on social action". This approach is concerned with seeing the world from the perspective of the research participants. Individuals are at the centre of the research process and people are actively constructing their social world, giving personal meaning to their situations and events, and making informed decisions to act in particular ways.

The interpretive approach was the choice for this study because it was concerned with descriptions that produce deep understanding and emphasised interpretation. Furthermore, it correlates to Terre Blanche and Durrheim (2002) who state that “ Interpretive research methods try to describe and interpret people’s feelings and experiences in human terms rather than by quantification and measurement”.

The research participants were educators who taught an object-oriented (OO) programming language. The researcher found the qualitative, interpretive approach suitable for the study because the aim of the study was to understand the teaching methods and learning experiences of educators and students

in tertiary and secondary institutions and how they constructed their relationships while experiencing OO programming. This meant studying educators in their ‘natural educational environment’ rather than under artificially created conditions.

3.3 FRAMES OF ENQUIRY

3.3.1 Qualitative research

As noted in the previous section, qualitative research was adopted for this study. The themes of qualitative enquiry are naturalistic, holistic and inductive.

The characteristics of the naturalistic theme of qualitative inquiry are listed below, together with their relevance in this study:

- Studies real-world situations as they unfold naturally.
- The study was conducted in the natural settings of a classroom/lecture theatre.
- Non-manipulative, unobtrusive, and non-controlling, openness to whatever emerges.
- The researcher did not influence the teaching strategy adopted by the participants nor dictate the implementation of one strategy over another.
- Avoids predetermined constraints on outcomes.
- The researcher did not put in place any notable constraints that would have affected the outcome of PP.

The theories that frame this study are threshold concepts and grounded theory.

3.4 THRESHOLD CONCEPTS

Meyer and Land (2005) have defined threshold concepts “as a way of describing particular concepts that might be used to organise the educational process”. Boustedt et al. (2007) claim that in Computer Science education there is “also a developing context for threshold concepts”.

Boustedt et al. (2007). During the educators’ and learners’ teaching and learning experience they would have encountered threshold concepts in Computer Science. The researcher is also keen to determine to what extent these threshold concepts may have influenced the sample population’s teaching and learning.

The purpose of this section is to discuss threshold concepts in Computer Science and reflect on their usefulness for this study.

Rountree and Rountree (2009) view threshold concepts in two parts: firstly as a model or framework, and secondly as “instance examples”. Threshold concepts provide academics with a model to develop their teaching and support student learning. The conceptual framework is intended to re-situate teaching and learning within the context of its own discipline, in contrast to the role which learning outcomes have developed as a management tool to audit and monitor ‘success’ (Hussey & Smith, 2003). In the traditional academic environment an educator would state ‘by the end of the lesson the student should be able to...’; however, in the threshold concepts environment learners go through a ‘transformation’, after which they gradually acquire the identity of the community of practice; they begin to “think more like a computer scientist” (Rountree & Rountree, 2009). During this transformation certain parts of the curriculum are pivotal: these represent the ‘portals’ that learners must traverse in order to succeed. To be considered a member of the community of practice, mastery of these concepts is required, and the process of mastery is seen as a sort of rite of passage.

Threshold concepts may be viewed as ‘instance examples’ and may be recognised by having (probably) all of the following five features: transformative, integrative, irreversible, bounded, and troublesome (Meyer & Land 2005).

Transformative:

- they change the way a student looks at things in the discipline;
- they create in the learner a new way of viewing and describing the subject and may alter the learner’s perception of themselves and the world.

Integrative:

- they tie together concepts in ways that were previously unknown to students; learners make new connections.

Irreversible:

- they are difficult for the student to understand;
- they are irreversible since the fundamental qualitative change that occurs is unlikely to be unlearned.

Troublesome for students:

- Meyer & Land (2003) state that “these concepts are problematic for learners because they demand an integration of ideas....., and this requires a transformation of a student’s understanding”.
- Perkins (2005) has termed the concept of “the underlying game” as an activity of “a student that may grasp concepts but the barrier to their learning appears to lie at a deeper level of understanding”.
-

Bounded:

- they indicate the limits of a conceptual area or the discipline itself;
- this “helps define the boundaries of a subject area because it clarifies the scope of the subject community” (Land,Cousin, Meyer & Davies, 2005)

The programming approach of procedural programming and OO programming complements the above threshold concepts. Programming concepts (selection, iteration, classes, objects, encapsulation and inheritance) are transformative, integrative, irreversible, and troublesome for students, and are boundary markers for them. Furthermore, the attributes of collaborative learning and PP and, in particular, the concept of ‘sharing knowledge’ acquired from the sample population are examples of the ‘potentially troublesome for students category’. Learners and educators were used to learning/working and being assessed on their own; PP and the subsequent sharing of resources/knowledge proved troublesome. This will be further elaborated upon in Chapter Five.

Land, Cousin, Meyer and Davies (2005) included threshold concepts for course design and evaluation, with nine considerations based upon these that they felt were important in the design and subsequent evaluation of curricula in higher education. These nine considerations are discussed in greater detail below.

3.4.1 Nine considerations for threshold concepts

The nine considerations for threshold concepts are jewels in the curriculum; the importance of engagement; listening for understanding; reconstitution of self; tolerating uncertainty; recursiveness and excursiveness; pre-liminal variation; unintended consequences of good pedagogy; and the underlying game or threshold conception.

- **Jewels in the curriculum**

By identifying threshold concepts as a “jewel in the curriculum” Land et al (2005) has given prominence to crucial point which an educator must identify and serve to reinforce before proceeding onto a new section in the curriculum. With respect to PP, crucial points are the rules that pair programmers must abide by – who would be the navigator/driver? An educator would be responsible for setting up the pair and providing motivation to continue PP. With respect to OOP an educator must ensure that a student can identify a class or an object.

- The importance of engagement

An educator should consider the specific forms of engagement which will be most appropriate to bring about particular transformative understandings at various points in the curriculum. Lather (1998) referred to these as “where the effort is to ... provoke something else into happening – something other than the return of the same”.

Land et al. (2005) couple this point of engagement and a teacher's need to provide a supportive liminal or transition environment.

- Listening for understanding

Teaching for understanding of threshold concepts needs to be preceded by listening for understanding. Ellsworth (1997) states that an educator must “cultivate a third ear that listens not for what a student knows but for the terms that shape a student’s knowledge”. With respect to PP it was noted that an educator cannot adopt a passive role in a PP scenario but rather an eavesdropping role and to intervene where necessary amongst paired partners.

R Land et al. (2005) referred to the term “pre-liminal variation: the ways in which students approach or come to terms with a threshold concept”. They state that one cannot “second-guess where students are coming from or what their uncertainties are”. If an educator has to adopt the “pre-liminal variation term” in a PP environment then very simplistically they will have to adopt the programming stance of their student and to “see” the obstacles and difficulties of working in a paired partnership and to learn a new programming concept.

- Reconstitution of self

This term refers to either the student or the educator “repositioning themselves in relation to their subject matter”. Curriculum designers may want to redesign troublesome concepts or reposition the knowledge within a curriculum. Meyer & Land (2003) state that “knowledge may be troublesome because it has become ritualized or inert, because it is conceptually difficult or alien, because it is tacit and perhaps requires awareness of an ‘underlying game’ imperceptible to the student, or because of the discourse that has to be acquired for the concept to become meaningful”.

Perkins , Meyer , and Land (2006) describe reconstitution of self as follows:

“as students acquire threshold concepts, and extend their use of language in relation to these concepts, there occurs also a shift in the learner’s subjectivity, a repositioning of the self ...”

What is being emphasized here is the inter-relatedness of the learner’s identity with thinking and language.

- Tolerating uncertainty

Land et al. (2005) discussed this point with reference to data collected from their respondents, and summarized their findings as “learners tend to discover that which is not clear initially often becomes clear over time”. They quote a media student who almost abandoned her studies but continued and eventually did cope with the threshold concepts. Efklides (2006) emphasised the indispensable role of metacognition in the learning process “both directly by activating control processes and indirectly by influencing the self-regulation process that determines whether the student will get engaged in threshold concepts or not”.

- Recursiveness and excursiveness

Land et al. (2005) argued that for the notion of learning as excursive, a journey or excursion which will have intended direction and outcome but will also acknowledge deviations and unexpected outcomes within the excursion; there will be digression and revisiting (recursion). Troublesome knowledge often requires revisiting and the “outcome of learning is not just the student will be able to but rather that the learner will have been transformed by the journey into one who thinks differently”.

- Pre-liminal variation

This consideration in the study refers to students negotiating the liminal place of understanding that is a state of confusion and anxiety with no apparent progress. Students present a partial, limited or superficial understanding of the concept to be learned, which R Land et al. (2005) have characterized as a form of ‘mimicry’. It is recognized that there will be a range of pre-liminal understandings that a cohort of students brings to the learning interactions. These are the concepts they carry into the liminal place where, in the case of PP, further concept development is mediated by peer discussion. The aim of this study was to investigate the extent to which PP assisted in developing a more clearly defined, useful set of post-liminal concepts. With this concept in mind, data collection and analysis focused on students' ability to work in pairs and gain confidence to verbalise learned concepts to each

other. These data were later extracted from the questionnaires, regarding students' preference for working in pairs or alone. It was further observed and noted in the data analysis that PP produced a greater degree of enjoyment.

- Unintended consequences of generic 'good pedagogy'

Educators are at pains to simplify concepts they perceive to be difficult for learners to understand. Sometimes their efforts to simplify exceed the scope of the concept to be simplified, and the meaning is lost, an experience supported by (Shanahan & Meyer 2006). With respect to PP it was noted by the researcher that in an effort by educators to adopt PP techniques, educators adopted a superficial and quick fix approach of implementing PP i.e. they simply paired off students and left them to their own means. Although well intended it proved dysfunctional. "What was traditionally thought of as good pedagogy may on occasion break down or prove to be dysfunctional in acquiring threshold concepts" (Shanahan & Meyer 2006).

- The underlying game or threshold conception

Perkins et al. (2006) defines an "underlying game" or "threshold conception" in instances where students are able to grasp concepts but the barrier to learning appears to lie at a deeper level of understanding. Meyer and Land (2005) quote an example in Computer Science programming, and this would highlight the relevance to this study: students may grasp the concepts of OOP, but may not appreciate the threshold conception of the underlying game of the interaction of all these elements in a process of ever-increasing complexity. Similarly with PP: it was noted in the data collection that students and educators alike may grasp the concepts of PP but do not appreciate its benefits and may actually sabotage the learning experience due to personality differences.

3.5 GROUNDED THEORY

Grounded theory is a research framework that was initially proposed by Glaser and Strauss in 1967. Using grounded theory a researcher commences with data and uses them to develop theory that provides relevant interpretations, applications, predictions and explanations (Glaser & Strauss, 1967). Grounded theory specifies an analytical strategy to collect rich data from multiple sources. It involves a process of collecting data to fill conceptual gaps, applying constant comparative analysis, refining concepts, defining the properties of the categories and identifying their relevant contexts (Boychuk, Judy, & Morgan, 2004) Grounded theory is an approach where theory is generated inductively from

analysis of the data as concepts are formulated into a logical, systematic and explanatory scheme (Strauss & Corbin 1998)

The methodology that formed this study's data analysis drew on Glaser's writing of grounded theory (Glaser, 2001; Glaser & Strauss, 2009).

Glaser & Strauss (2009) state that "Grounded theory offers a rigorous, orderly guide for theory development and is designed to allow the researcher to be free of the structure of more forced methodologies. Its real strength lies in its open-ended approach to discovery". The goal of grounded theory in research design in this study was to guide data collection from participants as they programmed alone and in pairs.

The technique that is central to grounded theory is coding. Coding refers to the analytical processes through which data are fractured, conceptualised, and integrated to form theory (Strauss & Corbin 1998). Charmaz (2003) and Leedy & Ormrod (2001) state that "Data analysis occurs through various coding procedures: open coding, axial coding, selective coding and development of a theory".

- Open coding

This is the first analytical coding process. Its purpose is to mark text or other informative data and to associate codes with the marked segments of text. Data are deconstructed in concepts and marked line-by-line and coded. This process of grouping concepts into higher levels of abstraction is called categorising and this forms the basis of grounded theory construction (Henning, 2004). The selection of groups for comparison highlights the various similarities and differences, which is vital for defining the different categories (Glaser & Strauss, 1967).

- Axial coding

This coding is defined as "the process of relating categories to their subcategories" (Strauss & Corbin 1998; Glaser & Strauss, 2009). Its purpose is to refine the information about each category and its subcategories, determining more about a category in terms of its conditions, context, strategies and consequences (Leedy & Ormrod, 2001).

- Selective coding

This coding is defined as “the process whereby a main category is selected to which other categories are related” (Henning, 2004). It is a process where categories are organised around a central explanatory concept where major categories are related (Strauss & Corbin 1998).

These coding procedures were used to guide the analytical process applied in Chapter Five.

3.5.1 Application of grounded theory to this study

Grounded theory was chosen for the data analysis in this study in order to facilitate the researcher’s need to answer the research questions based on the experience of learners and educators. In conducting this study the researcher was searching for influences that may have enhanced or hindered the teaching and learning experience of educators and learners. Much of this data analysis will form part of theory that future implementers of PP can take note of.

Data was collected from three high schools and two tertiary institutions. Learners and educators were exposed to PP concepts. The sample population experienced PP and this coincided with Strauss & Corbin (1998) who stated that “A key idea is that much of the theory development does not ‘come off the shelf’ but rather is generated or ‘grounded’ in data from participants who have experienced the process”.

This study is based on programming in an OO environment and as advocated by Havenga, Mentz, & de Villiers (2008). Grounded theory was used in Information Systems research, including areas such as OO programming, to synthesise a theoretical framework.

The subsequent steps of grounded theory, as noted in the previous section, are revisited in the section below and were applied in this study:

- **Open coding**

Data was deconstructed into parts. During the process of open coding the following questions were asked:

What did the respondent say when programming on his own/or with a partner?

What actions did the respondent take when confronted by difficulties in programming?

What did the respondent say when programming with a pair?

What actions did the respondent take when confronted by difficulties when programming with a partner?

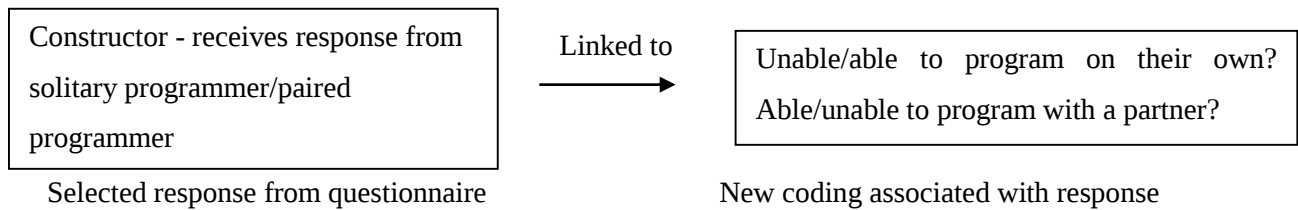


Figure 3.1: An example of selected text with associated new code

The 'code ability' to program with or without a partner was later categorised as to the level of satisfaction/frustration/ category as an indication of the application of PP in OO programming.

- Axial coding

Different categories are combined in new ways to make connections. The researcher focuses on categories that may be clustered together or subdivided into subcategories. Several closely related concepts can be organised into a major topic of interest. Furthermore, it may suggest dropping or adding a theme or examining other themes in more depth (Newman, 2012). For example, the theme improved and faster programming task completion is examined in more depth from an educator and learner perspective. The educator is able to complete the syllabus faster and learners work faster because programming tasks are shared and completed, in theory, in half the time.

- Selective coding

During selective coding a main category is selected to which other categories are related. Selective coding involves scanning data and previous codes and selectively looking for cases that illustrate themes (Henning & Van der Westhuizen, 2004; Newman, 2012)

The selection of groups for comparison makes similarities and differences distinct. For example, a coded main category named 'Benefits of PP' is a theme in Chapter Five and comprised of several different codes that related to aspects within this theme.

- Interpretation and development of a theory

Qualitative interpretation with a final thematic pattern is constructed from findings (Henning & Van der Westhuizen, 2004). During the process of interpretation different methods and actions all come together to motivate and defend the interpretations, and to develop an emerging theory on cooperative learning and PP processes. Chapter Five and Chapter Six give details on such emerging theory relating to participants' experiences in PP.

3.6 CONCLUSION

Threshold concepts and grounded theory form the theoretical basis of this study. As reported in the studies above and elsewhere, the programming concepts of OO programming complement the threshold concepts. The data analysis is discussed further in Chapter Five and Chapter Six. The next chapter details the methodology employed in this study.

CHAPTER FOUR

RESEARCH METHODOLOGY

4.1 INTRODUCTION

This chapter details the research process that led to collection of the data that were used to provide answers to the research questions. This chapter provides a discussion of the research context, research design, rationale behind the research methodologies and data collection instruments, data collection process, ethical considerations, sampling strategies and techniques that were used in the analysis and interpretation of data.

In order to gauge learners' and educators' perceptions of learning and teaching programming, questionnaires were administered, interviews conducted and lessons observed. The learners who were studying IT as a learning area had not previously been formally exposed to PP. Educators did, however, indicate that weaker pupils tended to pair off informally with brighter pupils, and that PP techniques were not discouraged. However, there was no formal introduction of PP techniques. The questionnaires, interviews and observations sought to determine educator and learner perceptions of their own programming strategies. The data collection sought responses in three areas:

- Use of a cooperative learning technique such as PP in teaching IT.
- Advantages and disadvantages of PP.
- Impact of pair versus individual programming.

4.2 RESEARCH CONTEXT

The research was conducted at two universities (campus A and B) and three secondary schools from KwaZulu-Natal (schools A, B and C). The researcher used convenience sampling, using a sample to which access was readily available. The rationale behind choosing the particular schools was the fact that they would be using a particular programming approach and thus the students' and educators' perceptions and experiences would be crucial for this study. The class size at the schools was small (averaged 10); in comparison the university tutorial class size was 20, and that at the university of technology the tutorial class size was much larger at 60.

The secondary schools and universities had adequate numbers of computers (each learner had a personal computer) and infrastructure (data projectors, laptops and even smart board technology at the university). All educators were adequately qualified and displayed a willingness to embrace the concepts of PP. The sample population is elaborated upon in greater detail in Chapter Five.

4.3 RESEARCH METHODOLOGIES

Having identified the research topic for this study, it was appropriate to develop research questions that would provide solutions and answers needed to address the research at hand. The research questions provided a focused means of investigating the research area (Bauer & Gaskell, 2000; Cohen, Manion, & Morrison, 2000)

The research questions were:

1. What are learners' experiences of solving programming tasks?
2. What teaching strategies are being used to teach Object-Oriented (OO) programming?
3. How does PP enhance problem solving in OO programming?

The first research question was aimed at finding out the learners' experiences of solving programming tasks. Research question two was aimed at determining how educators taught OO programming, and research question three targeted the experiences (both positive and negative) of educators and learners when using cooperative learning techniques in the classroom/lecture room.

The researcher chose to use a mix of qualitative and quantitative methods for this study. However, the emphasis was on qualitative methods and only frequencies from quantitative analysis were used to support the qualitative data. Qualitative research involves collecting textual or verbal data (data which cannot be counted), and involve observation of people, followed by careful description and analysis (Boeree, 2008). Qualitative research methodology is an inductive and exploratory tool and this was adhered to during the pre-test data collection phase. It is therefore an in-depth analysis of a problem in order to understand human behaviour (Hatch, 2002). Qualitative research is concerned with exploring social and human problems in a natural setting, with the intention of understanding what people feel and the experiences that have caused them to have these feelings.

Quantitative research, by contrast, is more highly defined and closely related to research in the physical sciences (Marais & Mouton 1988). Quantitative research involves collecting numerical data; the data from this type of research can be represented statistically and graphically. This type of research facilitates the analysis and comparison of data.

The benefit of using the mixed-methods approach is that it allows the researcher to capture the best of both the qualitative and quantitative approaches (Spicer 2004). Since this study required in-depth knowledge of learners' experiences, perceptions and attitudes when using a cooperative learning technique in an academic institution, the researcher chose to conduct this study within the mixed-

methods framework. Asking open-ended questions enabled the researcher to gain insight into the personal experiences of the respondents, and also assisted in obtaining information regarding how learners benefitted from use of a cooperative learning technique (Cohen et al., 2000). After much deliberation on the merits of the various data collection instruments, it was decided that it would be necessary to use more than one method in order to obtain data that would further the aims of the study and strengthen it. Observation allowed the researcher to observe the learners' interaction with PP strategies and concepts. The questionnaire enabled the researcher to gauge the experiences of learners in the use of PP (post-test) versus solitary programming (pre-test), and the interview enabled the researcher to gauge educator perceptions and experiences of using PP.

Table 4.1 indicates the data collection instruments used in the study.

Data collection instrument			
Research questions	Observation	Questionnaire	Interview
1. What are learners' experiences of solving programming tasks?	√	√	
2. What teaching strategies are being used to teach OO programming?	√	√	√
3. How does peer-to-peer learning enhance problem solving in OO programming?	√	√	√

4.4 TIMEFRAMES FOR DATA COLLECTION

The following timeframe was decided upon. The learners were introduced to programming concepts in the first term (February to March) at school/university. The observation process continued for the duration of the first term, for as long as the programming was being used in the classroom under the researcher's supervision. The pre-test questionnaire and the associated observations and interview process was conducted during the first part of the first term. Learners were then exposed to PP concepts. The post-test questionnaire and its associated observations and interview process was completed in the second term (April to June). The following sections discuss each of the data collection methods in detail.

4.5 OBSERVATION

Observational research involves the researcher making observations. This method of collecting data involves the researcher going into a classroom, school or university and observing what is actually taking place there. The researcher writes down a description of what he/she sees happening in the classroom (Cohen et al, 2000).

Learners were given activities involving programming tasks. These lessons were observed because the researcher wanted to gauge the learners' initial reaction to and interaction with a programming task. This was structured observation; the programming activity was video recorded, with a focus on the programming activity involving paired programmers. While the learners were engaged in programming tasks, the researcher informally observed them to monitor and observe their interaction with the programming task. The researcher's presence in the classroom as an observer was nothing out of the ordinary because of his dual role (lecturer and researcher). This suited the data collection method because the researcher wanted to remain as 'invisible' as possible in order to get an idea of the learners' experiences without interfering with their behaviour. The observation process also provided the researcher with realistic data that the learners may not have been able to provide through the questionnaire or interview, such as the learners' hands-on interaction with the programming compiler/user interface. See Appendix D for the observation schedule.

The advantage of observing the learners in the classroom was that data was collected when and where the activity was occurring, and the researcher could directly observe what the learners were doing. The disadvantage to using the observation process was that it was susceptible to observer bias – especially with the observer being both researcher and lecturer at the same time or during the video recording of the PP activity or when observing other educators in the classroom/lecture room/computer lab. This was achieved by using a mixed-methods approach, with observation being structured and the presence of the researcher being unobtrusive while observing.

4.6 QUESTIONNAIRES

To extend the research the participants completed a questionnaire about their programming experiences. The questionnaires were administered by the researcher and the educators. Questionnaires were used as pre-test and post-test instruments. Closed-ended questions were statistically analysed by means of descriptive statistics, while the open-ended responses were analysed by comparing the responses and unearthing common themes. The questionnaires appear in appendix A and appendix B.

4.6.1 Value of a questionnaire

Attitudinal surveys (in the form of a questionnaire) consist of a series of statements which respondents are required to answer within pre-set responses. This method is widely accepted and used to collect valuable data on learner perceptions and attitudes. The questionnaire facilitates the collection and analysis of numerical data that are structured and can be easily generalised and administered by someone besides the researcher (Bauer & Gaskell, 2000; Cohen et al., 2000).

This data collection instrument is not without its disadvantages, in that it is weighed down by the time taken to design, pilot and refine it (Cohen et al., 2000). A questionnaire is also limited in terms of the scope of the questions that can be asked and the range of responses that can be anticipated (Heather, Rollnick, & Bell, 1993). Owing to strict time constraints at the schools, university and university of technology, the data collection had to be quick and efficient while ensuring reliability and validity. A structured questionnaire met the needs as far as this was concerned, since the questions were pre-set and the responses fell within a prescribed range. Learners selected a response from those that had already been pre-set.

According to Cohen et al. (2000) a properly designed questionnaire facilitates the process of analysis, and this can be made even easier when the researcher is involved in the design. A questionnaire was useful in this context because it was not time-consuming to administer by the researcher or educators. The workload of the educators and learners coupled with the timetable at the schools and universities did not warrant a method that made further demands on educators' and learners' overburdened workloads. Closed-ended questions based on a Likert scale proved most productive.

All of the first-year IT students from the university of technology were invited to participate in the survey, in order to ensure that an adequate number of learners would respond to constitute a valid study. Out of approximately 400 first-year IT programming students, 326 questionnaires were completed and returned to the researcher.

With regard to the secondary schools, approximately 50 questionnaires were distributed to the Grade 11 IT learners, and 45 questionnaires were completed and returned. At the university 43 questionnaires were completed and returned.

All responses were used to ensure that the data were not skewed; however, incomplete and invalid responses in the questionnaires had to be excluded. Detailed analysis is provided in Chapter Five. Despite the fact that the number of males and females was not equal, it was decided that the statistics

reflected the reality of the situation and that the status quo should remain. The questionnaire sought to measure learners' experiences with regard to the use of cooperative learning strategies.

4.6.2 Development of the questionnaires

Development of the questionnaire was guided by an extensive review of the literature. The questionnaire included questions eliciting the basic views of respondents towards computer programming in the classroom, their experiences of cooperative learning strategies and actual support for PP.

Learning-related and collaboration questions were posed to assess the impact and context of cooperative learning strategies and the adoption of PP concepts by educators and learners. The questionnaire asked questions based on Salleh (2008) "Protocol for Systematic Review of Pair Programming". Mentz et al. (2008) posed similar questions in the study of incorporating learning principles in PP for students. Since the questions in this study had been used in a similar context (Padmanabhuni et al., 2012), the researcher felt validated with regard to the acceptance and relevance of the data collection instruments used and adapted for this study.

Havenga, De Villiers, and Mentz (2011) developed their original questionnaires from an exploratory analysis of literature pertinent to the cooperative learning and teaching of programming. Their questions were taken directly from previously validated research. A similar approach was adopted in this research, where measures were taken to elicit the direct determinants of the PP in a classroom context: collaborative learning and PP, benefits of PP, and shortcomings of PP. Questions were derived for each of the direct determinants.

4.6.3 Rating scale

Questionnaires frequently make use of rating scales to determine behaviour and opinions. The Likert scale originates from Renis Likert, who developed this technique for the assessment of attitudes (Likert, 1932). The Likert scale caters for a range of responses that exhibit varying degrees or intensities of feeling, making it a flexible tool to use, and yet it enables one to generate statistics that can easily be analysed. The present questionnaires employed a Likert scale on the continuum of Never, Seldom, Often and Always, using a 4-point scale so as to avoid any middle option. This was used to indicate educators' and learners' level of agreement on PP concepts. Data from the learners

and educators were gathered to determine influential factors perceived by learners in PP and educators who adopted PP teaching strategies in their classroom.

Learners were asked to tick the appropriate column, and educators were asked to insert a '1' in the appropriate column. This facilitated the recording of responses on a spreadsheet that was used in analysis of the questionnaire. The educators had 22 questions on the Likert scale rating. Positive and negative statements were included in the scale.

4.6.4 Data collection method

Letters of consent (Appendix E) requesting learners' participation in the research study were given to the high school learners and their parents. Questionnaires (pre-test) were distributed by the research and/or the educators, and collected for analysis. Later educators were briefed on using PP techniques and used those of navigator and driver in their teaching strategy. At the University of Technology paired programmers were video recorded. Post-test questionnaires were distributed and collected for later analysis.

4.6.5 Analysis

The data from the questionnaire were captured on a spreadsheet in terms of actual numbers. This facilitated the statistical representation of data in terms of percentages and graphs. Data from the questionnaire were captured on Microsoft Excel. The questionnaire was summarised with learner responses under the following headings: Never, Seldom, Often, and Always. In the next chapter the researcher uses percentages to indicate learner responses.

4.7 INTERVIEW

After the observation and questionnaire, the third method of data collection was interviewing. Interviewing is a technique that employs questioning as its principal method of data collection (Leedy & Ormrod, 2001; Marais & Mouton 1988; Terre Blanche & Durrheim 2002).

As a data collection method the interview may vary from those that are completely unstructured to those that are completely standardised and structured (McMillan & Schumacher 1993; Cohen et al, 2000; Johnson & Waterfield, 2004). Seidman (1998) points out that the basis of interviewing is the desire to understand other people's experiences and what they make of such experiences. He states:

“provides access to the context of people's behaviour and thereby provides a way for researchers to understand the meaning of that behaviour. A basic assumption in in-depth interviewing research is that the meaning people make of their experience affects the way they carry out that experience...Interviewing allows us to put behaviour in context and provides access to understanding their action” (Seidman 1998, p. 4)

The researcher chose to use a semi-structured and standardised open-ended interviewing method, because this was a powerful way of gaining insight into educational issues and hence would give the researcher and respondents the opportunity to explore and discuss issues together, face-to-face (McMillan & Schumacher 1993; Seidman 1998). This type of interviewing is in line with the sequential inter-method- mixing technique (or method triangulation), which is in keeping with the mixed-methods approach/mode employed in this study (McMillan & Schumacher 1993).

Henning (2004) described standardised interviews as a data production method in which the interviewer is to control the process so as to ensure that the interviewee does not wander off the topic, yet allowing the respondent(s) to ‘freely’ give subjective answers (that yield information that represent reality more or less as it is through the response of the interviewee) to the questions posed by the interviewer. Thus the interview method employed in this study took the form of a standardised open-ended interview which used semi-structured questions (McMillan & Schumacher 1993; Terre Blanche & Durrheim 2002). All the interviews were guided by a set of questions (refer to Appendix A & B), and were recorded using a digital voice-recorder and later transcribed (available electronically).

4.7.1 Sampling and data collection

The interviews were semi-structured and conducted with a sample of six participants. Participants in the interview were IT educators from secondary schools, a university of technology and a university. Dates for the interviews were arranged and fixed, and scheduled for the last week of June 2012 and August 2012. All interviews were recorded and later transcribed.

4.7.2 Design of the interview

A list of interview questions was drawn up to direct the interviewer and the interview process. See Appendix F.

4.7.3 Data collection method

An interview timeline was drawn up indicating dates and times during which educators would be interviewed. Educators were contacted via email and were told a day in advance when they would be interviewed. Copies of the interview questions were emailed to the educators prior to commencement of the interview, so that they had time to peruse them. Educators were told that they could jot down their thoughts and ideas if they wished to. They were not compelled to write down responses on their interview sheet. At the start of the interview the researcher welcomed and thanked the educator and then explained the research process. The researcher reiterated their rights as participants and outlined the interview process to them. Educators were informed that the data collection from this interview would be analysed and used in the write-up of the dissertation, while respecting the confidentiality of the individual interviewees.

4.7.4 Analysis

The analysis of data encompasses the breaking up of complex data into manageable themes, patterns, trends and relationships (Marais & Mouton 1988). Analysing what the respondents have said in an interview requires the researcher to relive the interview and to link the responses with the underlying theories, while looking for evidence to support or contradict them (Bauer & Gaskell, 2000). This part of the analysis began with producing the transcript of the IT educators' interviews. The researcher preferred to do the transcription himself because the voices could be easily recognised and he could recall what was said if the recording was unclear, since the interviews were still fresh in his mind. It also gave the researcher a chance to relive the interview process by going through every word and expression in an effort to try and make sense of the data. Once the transcripts were done, the researcher looked for themes and categories that were associated with the theoretical framework and the research questions.

4.8 Validity/reliability/trustworthiness

The application of a multi-methods approach allowed for a comparison of data – referred to as triangulation (Krefting, 1991). According to Denzin and Lincoln (2000) triangulation is a means of ensuring concurrent validity and prevents personal bias. Validity refers to “the appropriateness of the conclusions claimed from the analysis of the collected data” (McMillan & Schumacher 1993; Terre Blanche & Durrheim 2002). This has to do with whether the research methods, approaches and techniques used were appropriate for the study conducted. To ensure credibility in this study, the

researcher interviewed (using a voice-recorder) the participants with the intention of gaining insight into their understandings and experiences of the learning benefits and challenges they were facing with integration of PP into teaching and learning. The interviews were transcribed. Data were also collected from observation of lessons using the structured observation schedule and the questionnaires. The collection of data from different sources added to the strength of the validity, reliability and trustworthiness of the study.

4.9 ETHICAL CONSIDERATIONS

According to McMillan and Schumacher (1993) “ethical issues refer to all the precautions, steps and efforts that researchers carefully put into practice to protect the research participants while interacting with them for data production”. Bell (2005) argues for the establishment of ethics committees which can ensure that no badly designed or harmful research is permitted. A credible research design involves the selection of participants, effective research strategies, and ensuring that all of the steps of the research adhere to research ethics.

During the planning and implementation of this research project consideration was given to ethical issues relating to using learners as part of the data collection method. The researcher applied for ethical clearance from the University of KwaZulu-Natal (Appendix J) and the Department of Education (Appendix H) to conduct this research at the schools. In the application the researcher outlined the type of research that was going to be done, the research methods and the data collection instruments that were to be used. The application also included how ethical issues concerning participants were to be addressed. Once ethical clearance was issued, the data collection process began. The school principals granted the researcher permission to use the school as the research facility, in accordance with ethical guidelines that were presented to them. Learners were assured that they were not compelled to participate in this research project.

The researcher first had to get permission from the adult under whose authority the learner was during the context of the research, and secondly from the learners. Letters of consent were sent to parents of all Grade 11 IT learners (Appendix E). The consent form outlined the research title, including its broad aims and purposes. The consent form also assured participants of absolute confidentiality; this is a very important aspect in getting participants to answer truthfully. Seeking consent was necessary as it protects both the learner and researcher from any problems that may arise, and also provides proof of the authenticity of the data collected and the processes used Cohen et al. (2000).

4.10 LIMITATIONS OF THE STUDY

This study was conducted at the Computer Science departments at a University and a University of Technology, and at three high schools in KwaZulu-Natal; therefore the findings of this study cannot be generalised since the sample is small. Furthermore, the IT curriculum is rigid and IT educators generally use prescribed textbooks that adopt a particular programming approach (either PPA or OO programming); therefore this study may also be limited by the methods adopted at a particular learning institution.

The following aspects were not the focus in this study:

- programming paradigms other than OO programming;
- differences in programming language constructs;
- the curriculum at high schools and tertiary institutions.

Furthermore, the study at secondary schools involved Grade 11 and Grade 12 IT learners, whose responses could not represent the entire learner population at the school. A further limitation of this study is that participants may not have been totally honest with the researcher, for various reasons such as shyness or wanting to protect privacy. Participants may have given responses which they considered appropriate, but which may not have been true or valid (De Vos, Strydom, Fouche, & Delport, 2002). This was particularly true of responses from educators, who sought to defend their teaching strategy and justify reasons why they could either implement PP or not. The results of this study could not be generalised.

4.11 CONCLUSION

This chapter discussed the methodology used in the study. It also discussed the instruments used in the data collection: questionnaires, interviews and informal observation. Ethical considerations were also discussed. The advantages and challenges accompanying the use of mixed research methods were highlighted, and the limitations of the study were also discussed.

The next chapter presents, analyses and interprets the findings of the study.

CHAPTER FIVE

DATA ANALYSIS AND INTERPRETATION

5.1 INTRODUCTION

Chapter Five provides a summary of the analysis of data gathered during the research process. This analysis presents data and arguments from the observation, interviews and questionnaires used to collect data. The questionnaire used a Likert scale to determine learners' and educators' attitudes towards solitary programming versus PP. Data from the questionnaires were captured on Microsoft Excel. The data from the questionnaire were summarised with learners' responses placed under the headings: Never, Seldom, Often and Always. In the discussion in this chapter percentages are used to indicate learners' responses. Data collected from the responses were analysed using SPSS version 20.0.

Learning-related questions, interactive questions, and PP-related questions were employed to assess the impact and context of the adoption of PP techniques by learners and educators.

The results are presented in the form of graphs, cross-tabulations and other figures.

In addition, this chapter sets out to answer the research questions:

- 1) What are learners' experiences of solving programming tasks?
- 2) What teaching strategies are being used to teach OO programming?
- 3) How does PP enhance problem solving in OO programming?

The chapter further:

- implements the research design and methodology outlined in Chapter Three and Chapter Four as it considers interpretation of participants' PP experiences or the lack thereof;
- presents the results and discusses the findings obtained for the pre-test and post-test questionnaires from learners, the interview schedules from educators, and the observation schedules in this study; and
- discusses the analysis of empirical materials as the raw data are converted into final patterns of themes and categories using threshold concepts and grounded theory.

The study employed specific strategies to ensure that the main questions were answered by the data collection and analysis processes, which are summarised below.

Data collection techniques included computer programming: all pre-test participants were required to program on their own and then to complete a questionnaire which focused on this experience of solitary programming. Similarly, all post-test participants were required to program in pairs and to

complete a questionnaire. Educators were interviewed and observed in their programming classrooms and data were collected as per observation and interview schedules (Appendix D & F).

In terms of data analysis techniques, the data from the pre-test and post-test questionnaires were analysed by specific measurement criteria, as indicated in Table 5.1. The data were analysed with the aid of SPSS and where applicable, a description of certain statistical measurements is given.

Finally, coherence between different data sources was investigated to identify patterns of meaning and to describe the emerging themes in order to explain specific patterns and phenomena.

Triangulation was applied whereby structured observation data collection techniques and questionnaires from educators and students were used to analyse participants' written responses.

5.2 APPLYING GROUNDED THEORY

In this section the concepts of grounded theory are applied to the data analysis acquired during the data collection phase.

As per the discussion in Chapter Three, three major stages characterise constant comparison analysis (Strauss & Corbin 1998). During the first stage (open coding) the data are gathered into small units, and the researcher attaches a descriptor or code to each of the units. During the second stage (axial coding) these codes are grouped into categories. In the third and final stage (selective coding) the researcher develops one or more themes that express the content of each of the groups (Strauss & Corbin 1998)

Data from the pre- and post-test questionnaires were organised into small units. During selective coding certain patterns and themes emerged that appeared to answer the research questions. A category refers to a concept that represents a phenomenon (Strauss & Corbin 1998), while a theme represents a "chunk of reality" that can be used as the basis for an argument (Henning et al., 2004).

Pre-test and post-test data, observations of educators and students and the interview sessions with the educators were analysed via constant comparison analysis. This allowed the researcher to assess saturation in general and in particular across the pre- and post-test groups. The pre- and post-test data were analysed separately and later combined according to the categories that emerged.

The qualitative analysis involved interpretation of the students' programming abilities both as a solitary programmer and in PP. Selections from the participants' responses were assigned codes, which were organised into categories.

The following defining features of grounded theory were explained in Chapter 3, and Table 5.1 summarises how each related to this study.

Table 5.1: Defining features of grounded theory (Creswell, 2013, p. 85) and how each related to the study

1	Researcher focuses on process or an action
	In this study the researcher focused on the students' self-management of programming (pre-test) as compared to the paired management of programming (post-test)
2	Researcher seeks to develop a theory of this process or action
	The action related to implementation of PP and eventually to support learners and educators to successfully implement PP
3	Memo-ing becomes part of the developing theory as the researcher writes down ideas as data are collected
	This was done during the observation and interview phases of data collection
4	The primary form of data collection is often interviewing - here the researcher is constantly comparing data gleaned from participants about the emerging theory
	In this study the researcher interviewed educators and compared data gathered from the pre- and post-test results from learners
5	"Data analysis can be structured and follow the pattern of developing open categories, selecting one category to be the focus of the theory and then detailing additional categories (axial coding). The intersection of the category becomes the theory (selective coding). This theory can be presented as a diagram, as propositions, or as a discussion" (Strauss & Corbin 1998)
	Themes that emerged were student self-management of programming, paired management of programming and educator perspectives of programming

Glaser and Strauss (1967) cited four criteria for a well-constructed grounded theory.

Table 5.2 summarises how each of these features relates to this study.

Table 5.2: The four criteria of a well-constructed grounded theory, and how each relates to this study

1	Fit: The categories and properties should fit the realities being studied
	Categories that emerged were self-rating of programming, programming experience, and likeability/enjoy ability of programming
2	Work: In order to work the theories should explain variations in behaviour
	Variations in behaviour were programming in solitary and then programming in a pair
3	Relevance: This is achieved when a grounded theory both fits and works
	Grounded theory is relevant to this study as explained in Chapter Three, Table 5.1 and in analysis of the data in following sections. It both ‘fits’ and ‘works’ with the data
4	Modifiability: The emerging theory is open to adaptation as new data are integrated
	New data were integrated from the interview schedules with the educators and the observation; furthermore, as a recommendation for future study, assessing pair programmers for test and examinations could modify and adapt the findings of this study

Saturation did not occur until the end of the post-test and the data collection from the educators. The above themes were identified and generated from the statistics software in conjunction with the literature reviewed in Chapter Two.

5.3 APPLYING THRESHOLD CONCEPTS

Meyer and Land (2005) view threshold concepts as ‘instance’ examples and be recognised by being (probably) all five of the following: transformative, irreversible, integrative, bounded and troublesome. (Refer to Chapter Three for a detailed description of threshold concepts.) The questionnaires were used to operationalise the threshold concepts of Meyer & Land (2005).

5.3.1 Transformative

Not only did the introduction of PP ‘change’ the way in which the students viewed computer programming, it also changed the way educators approached the teaching of programming concepts. Educators realised the value of ‘pairing students’, thereby creating more time for them to carry out administrative tasks; a student would first ask their paired partner for help, and only when they had

exhausted their resources would they approach their educator. When students experienced difficulty 61% indicated that they would “consult teacher/fellow classmate” in the pre-test data collection, compared to 40% who indicated “consult with teacher” in the post-test data collection.

Students also indicated that by implementing PP they perceived programming differently. As discussed under the ‘likeability category’, more students indicated that they enjoyed programming and more students indicated that they would ask their paired partner for assistance. Almost 25% of respondents indicated that one of the reasons for enjoying programming with a partner was the fact that they would “assist each other”, and almost 17% indicated that they “share knowledge”. With reference to achieving their programming solution, almost 58% of respondents indicated that they “always achieved their programming solution with a paired partner”; furthermore, 54% indicated that their programs were of a higher quality when working as part of a PP partnership.

5.3.2 Integrative and troublesome for students

Initially learners were unaware that their informal methods of working together on a programming task could be formalised into a programming strategy. PP offered students an opportunity to present to their educators a programming solution which both partners had worked on; however, due to the competitiveness of these students since they were previously assessed on an individual basis, they were reluctant to embrace the assessment component of programming. Similarly, educators expressed concern with respect to assessments. Consider the following direct quotations from educators:

“Assessment is on an individual basis”.

“Programming is being critiqued and I don't have time to waste, so I don't really implement it”. [‘It’ being PP.]

The above two factors (competitiveness of students and assessments) correlated to the ‘Troublesome for students’ characteristic of threshold concepts.

According to Perkins (2005) this was defined as the “underlying game” or a threshold conception. Observational data (points 1, 2, 9 and 10) indicated that the sample population's ‘barrier to learning’ was ineffective sharing and application of PP strategies and the individual assessment of programming tasks.

However, to introduce a new programming concept and speed up the pace of lessons, 3 of the 5 educators agreed that PP was effective, as per interview schedule question 8 (refer to Appendix F). Hopefully in the future educators will correlate assessments and PP.

5.3.3 Irreversible

Initially in the pre-test only 1.1% of respondents indicated that “team work is encouraged”. Once students and educators experienced the benefits of PP, many indicated that they would consult fellow students (18%). What was strange was that only 0.5% of the respondents indicated that they would consult the learning material. Also, initially 60% indicated that they would consult their educator; after the post-test only 40% indicated that they would consult their teacher. This change in behaviour indicates to the researcher that students have attained the characteristic of a threshold concept being ‘irreversible’ and unlikely to be unlearned.

5.3.4 Bounded

Boundary markers relate more to disciplinary boundaries than individual differences (Boustedt et al., 2007). In echoing the characteristic of ‘bounded’ with respect to threshold concepts and PP, it suffices to state that analysing students programming on their own and with a paired partner provides a set of ‘boundary markers’ that educators can define and use as a strategy in their teaching.

5.4 ANALYSING DATA ACCORDING TO THE NINE CONSIDERATIONS OF THRESHOLD CONCEPTS

Land et al (2005) included nine considerations of threshold concepts (see Chapter Three, 3.2.1), and similarly this section includes these considerations for evaluation of the data acquired.

5.4.1 Jewels in the curriculum and Importance of engagement

Not only are the threshold concepts of PP ‘powerful’ transformation points in the students’ learning experience, they are also powerful transformation points in the educators’ teaching experience. It was noted that during the data collection phase with the educators, when questioned one educator stated “No comment, did not really implement paired programming” (refer to Appendix F). Furthermore, the observation schedule indicated that almost 80% of educators did not use PP, and 60% did not feel confident using PP concepts.

In order for an educator to use the PP concepts in their teaching they must not only embrace PP, they must also actively engage students by providing specific activities that encourage the sharing of

knowledge (Lather, 1998); as Chapter Three, section 3.2.1 states, this is to provoke something else into happening instead of the same.

Educators should also ensure that the learning environment is conducive to engaging PP. Data from the observation schedule indicated that 80% of the classrooms were not conducive for PP. Computers were laid out in threes and very close to each other, meaning that students'/pairs' conversations impacted on each other. If in the opinion of the educator the pairs are not actively involved or PP does not seem to be working, he/she may consider rearranging the pairs or disbanding the pair altogether. This leads to the next concept of 'listening for understanding'.

5.4.2 Listening for understanding

Apart from the fact that educators must embrace and engage with PP concepts, they must be actively 'listening for understanding'. During the observation phase, students actively questioned the educators, and later when they were more confident in their pair they questioned their partner. Listening for understanding also included non-verbal attributes, i.e. student pairs who were particularly quiet during programming or those with one dominant partner. This attribute of threshold concepts was also evident in the following observations:

Paired programmers conversed in their mother tongue and shared dialogue regarding a syntax or runtime error.

Paired programmers also acknowledged that at times the conversation wavered, and pairs spoke about irrelevant dialogue and wasted programming time.

5.4.3 Reconstitution of self

With reference to Chapter Three, section 3.2.1, grasping the threshold concepts of PP involves a repositioning of one's self in relation to programming. The data gathered indicate that the overall picture was that both students and educators had begun to think, act and identify themselves as part of a PP team or strategy. There has been a 'reconstituting' of programming as a solitary exercise, to programming as part of a pair. Most encouraging was that four out of the five teachers indicated that PP did indeed help their learners to better learn programming concepts (refer to Appendix F).

5.4.4 Tolerating uncertainty

To follow on from the statement in 5.5.1, one of the five educators indicated that they were not competent in using PP and that they did not “really implement paired programming”. Similarly, a small percentage (less than 2%) indicated “uncertainty of the implementing of PP”, citing “creates confusion and creates conflict” (refer to Appendix F)

The aforementioned data suggest that both the educator and students were uncertain in their attempts at implementing PP. Shanahan and Meyer (2006) also alluded to this point (refer to Chapter Three, section 3.2.1) and this was termed “liminal space”. They encouraged educators to use threshold concepts as an analytical tool to better understand learning in their subject and thus improve education.

5.4.5 Recursiveness and excursiveness

PP affords students the opportunity to work at their own pace and to revisit ‘troublesome concepts’. In this study the researcher also used the concepts of recursion and excursiveness. The educator’s role was that of a facilitator, and at the end of the lesson different pairs of programmers within a classroom were at different phases (excursions) in their understanding and comprehension of a particular programming concept. However, during the data collection phase observations revealed that time constraints, syllabi completion and looming deadlines dictated adoption of the linear teaching strategy rather than a recursive and excursive strategy.

5.4.6 Pre-liminal variation

As per Chapter Three, section 3.2.1, pre-liminal variation refers to a student's negotiating of the liminal place of understanding that is a state of confusion and anxiety with no apparent progress. With respect to correlating this with the data emanating from this study, it is pertinent to refer to the pre-liminal variation of both the student and educator. The students and educators experienced anxiety and confusion when implementing PP. Educators were unsure of ‘how to pair off students’ and students were equally anxious about the person that they were partnered with. Both educators and students expressed concern about the assessment criteria, but this factor was not part of the research questions or criteria. Overall, both students and educators reached consensus as to the positive benefits of implementing PP. Therefore PP can be viewed as a threshold concept which provided both

students and educators with a strategy when confronted with a new programming task or concept to either teach or learn.

5.4.7 Unintended consequences of good pedagogy

In an effort to subscribe to the concepts of PP for the study, educators and students alike simplified the processes of pairing by spontaneously and haphazardly forming pairs of programmers and thereafter coding. This proved to be ‘dysfunctional’ as per (Meyer & Land, 2003) (Chapter Three, section 3.2.1), and consequently would have created a barrier to implementation of PP. It was observed that due to time constraints educators initially felt that by implementing PP they would be unable to complete the allocated tasks and hence fall behind in their syllabus. However, when implementing the principles of PP it was noted in the interview schedule that three out of the five educators indicated that the use of PP ‘speeded up’ the pace of the lessons. As one educator stated:

“It halved the contact time with my students”.

5.4.8 Underlying game or threshold conception

As noted in section 5.5.8, increased administrative workloads and educators’ and students’ resistance to implementing a new strategy to teach and learn programming meant PP educators and students alike succumbed to the “underlying game” as per (Shanahan & Meyer, 2006) (section 3.2.1) of grasping a threshold concept of PP. However, the barrier to implementing appears to lie at a deeper level. This ‘deeper levels may be seen in the following responses from an educator to question 12 of the interview schedule:

“Yes – too much administration work; language barrier of students; not enough resources like paper; get work from other teachers/office to do in-between teaching times”.

The underlying barriers appear to be the increased administrative workload of educators, poor time management of educators, lack of basic resources, cultural differences of students, etc. This is above and beyond the scope of this study.

In terms of pre-test questionnaires, 222 were collected. All participants (n=215) were also required to complete the pre-test questionnaire (Appendix A) and to rate themselves as computer programmers, and 191 post-test questionnaires were collected. The overall reliability score for the ordinal section that comprises this construct was 0.667, which is close to the acceptable value of 0.70. This implied that the respondents scored the construct consistently.

With regard to gender, nearly two-thirds (62.6% pre-test and 63.5% post-test) of the sample was male, consistent with the literature reviewed (Chapter Two), where it was noted that IT programming tends to be dominated by males. The issue of gender was outside the scope of this study. However, the literature reviewed and observations of learners indicated that females tend to be more conversational when programming as a pair, and this outcome should be emulated by male programmers. The reason and rationale as to why there are so few female programmers is an opportunity for further study, and will be highlighted in Chapter Six. For the sake of convenience, from now on all participants will be referred to in the male gender.

Approximately half of the students were at first-year level. Even though IT students at third-year level completed the questionnaires, they were in effect introductory programmers who had no programming experience. A little more than two-thirds (68.5%) indicated that they did not have any programming experience, while approximately one-third (31.5%) of the respondents indicated that they had previous programming experience. Half of this grouping was at first-year university level.

In total, half of the respondents who did not have programming experience were at university. Those in second and third year who indicated that they had no previous programming experience had referred to programming experience at school.

Of the school learners, 18.3% indicated that they had not had programming experience, and the equivalent value for the university student was 50.2%. In total, 68.5% of respondents indicated that they had not been exposed to programming prior to doing a computer subject. Java appeared to be the most common language across the sample population. On average almost 74% of respondents were exposed to Java

Initially, in the pre-test, a little more than half (56.1%) of the respondents indicated that they believed that they were good programmers. A small percentage (3.6%) rated themselves as being advanced. The remaining respondents indicated that they were experiencing difficulty. In the post-test more than two-thirds of the respondents indicated that they believed that they were good programmers (68.6%). A small percentage (8.3%) rated themselves as being advanced.

The pre-test data indicated that only about one-fifth (19.9%) of respondents preferred working alone. More school learners preferred to work alone (24.1%) than university students (18.4%). Overall similar numbers of respondents indicated that they would be comfortable with working in a group or with a partner. More university students preferred working in a group, whilst more school learners preferred working with a partner. Again, in the post-test only about one-fifth (21.6%) of the respondents preferred working alone. Almost 80% of the learners appeared to be enjoying programming. This augers well for the overall performance on the course and correlates to the 'likeability factor' of the course.

Table 5.3: Indicating responses to a question about whether they enjoyed working with a partner

	Group		Total %
	School learner	University student	
Yes	84.6	87.1	86.8
No	15.4	12.9	13.2

At least 61.1% of the respondents in the pre-test indicated that they would consult with the teacher and their classmates; however, when posed with the identical question in the post-test, only 40.1% of respondents indicated that they would request the educator's assistance. It is evident that PP reduced educator workload and consultation time as per the literature (see 2.4.5).

Only 1.1% of respondents indicated that 'team work is encouraged'. This highlighted and confirmed the statement made in Chapter One. As IT educators, who require students to program individually during the teaching and learning process, should we not take our cue from the software industry which requires programmers to program in pairs/teams, and hence boost our IT learning outcomes and pass rates? A detailed description of the educator responses and a table depicting the observation schedule is included in Appendix D & F.

5.5 THEMATIC ANALYSIS

Analysis of the data was guided by grounded theory and threshold concepts, including transcribing and identifying themes. The section below provides key concepts derived from the theoretical framework, which were looked at in detail by the researcher during the classroom observation and interview processes.

The themes emerged from the pre- and post-test questionnaires, classroom observations and interviews with the educators, and were as follows: self-rating of programming, programming experience, and likeability/enjoy ability of programming. The educator's role referred to the processes of implementing, maintaining and ultimately terminating the PP strategy in the classroom and the student's role referred to the self-management of paired programming and how it influenced the successful adoption of the PP strategy.

5.5.1 Themes from the data analysis

5.5.1.1 Self-rating of programming

All participants were asked to rate themselves as computer programmers. Table 5.4 indicates the self-rating of the respondents in the pre-test.

Table 5.4: Self-rating of respondents, pre-test (%)

	Group		Overall
	School learners	University students	
Advanced	3.4	3.7	3.6
Good	54.2	56.8	56.1
Struggling	42.4	39.5	40.3

A little more than half of the respondents indicated that they believed that they were good programmers (56.1%). A small percentage (3.6%) rated themselves as being advanced. The remaining (40.3%) indicated that they were experiencing difficulties with programming. They felt that programming was complicated and that they had insufficient knowledge; this was, amongst others, the main cause for experiencing difficulties in programming.

Table 5.5 indicates the self-rating of the respondents as a computer programmer from the post-test results.

Table 5.5: Self-rating of respondents, post-test (%)

	Group		Total
	School learners	University students	
Advanced	0.0	9.6	8.3
Good	73.1	68.0	68.6
Struggling	26.9	22.5	23.0

More than two-thirds of the respondents indicated that they believed that they were good programmers (68.6%); in the pre-test an average of 55% did so, so this indicates a definite increase. A small percentage (8.3%) rated themselves as being advanced. The remaining respondents (23%) indicated that they were experiencing difficulties with programming, which is a decrease from the pre-test figure of 40.3%. When comparing whether respondents achieved their programming

solutions by working on their own or within a paired partnership, a similar result was achieved. Almost 60% of respondents indicated that they often achieved their programming solutions when working on their own; similarly, in the post-test almost 58% indicated that they did so whilst working with a partner.

Table 5.6 indicates the advantages of using a paired partner to program.

Statement	Rating	%
We achieve our programming solution	Often	58
When I write programs with a fellow student the programs are of a higher quality	Often	54
Our programs usually have syntax, run time and logic errors	Seldom had	54
We usually obtain a programming solution within the allocated time period	Often	47
Working as a pair we share each other's frustration when we are unable to successfully compile our program	Always	49
We have the necessary problem-solving skills to find a solution to a programming task	Often	56

Table 5.6: Figures from this study regarding the advantages of using a paired partner to program

Of particular importance was the statistic that 54% of the respondents felt that their programs created using PP were of a higher quality and had fewer syntax and logical errors. During the post-test respondents indicated that they could share their frustrations when faced with a non-compiling program or inability to grasp a particular programming concept; in contrast, when programming on their own respondents only had their educator to re-explain the programming concept or to find the error in their non-compiling error-prone program.

5.5.1.2 Programming experience

A little more than two-thirds (68.5%) of respondents indicated that they did not have a programming background, while approximately one-third (31.5%) indicated that they had previous programming experience. This grouping included those in their first year at university. In total, half of the respondents who did not have programming experience were at university. Those in second and third year who indicated that they had no previous programming experience referred to experience gained at school. Of the school learners, 18.3% indicated that they did not have programming experience; the

equivalent value for the university students was 50.2%. In total 68.5% indicated that they had not been exposed to programming prior to doing a computer subject.

On average, 34% stated that they had no programming experience. This statistic had the following effects on PP:

- The more experienced programmers tended to dominate the PP experience.
- Educators found it difficult to pair programmers because of differences in programming abilities and experience.
- In PP the weaker, slower and more inexperienced programmers did benefit from working with more experienced programmers.

5.5.1.3 Likeability of programming

Pre-test likeability factor

Figure 5.1 shows percentage responses to ‘Do you like to solve programming problems by working alone or within a group or with a partner?’

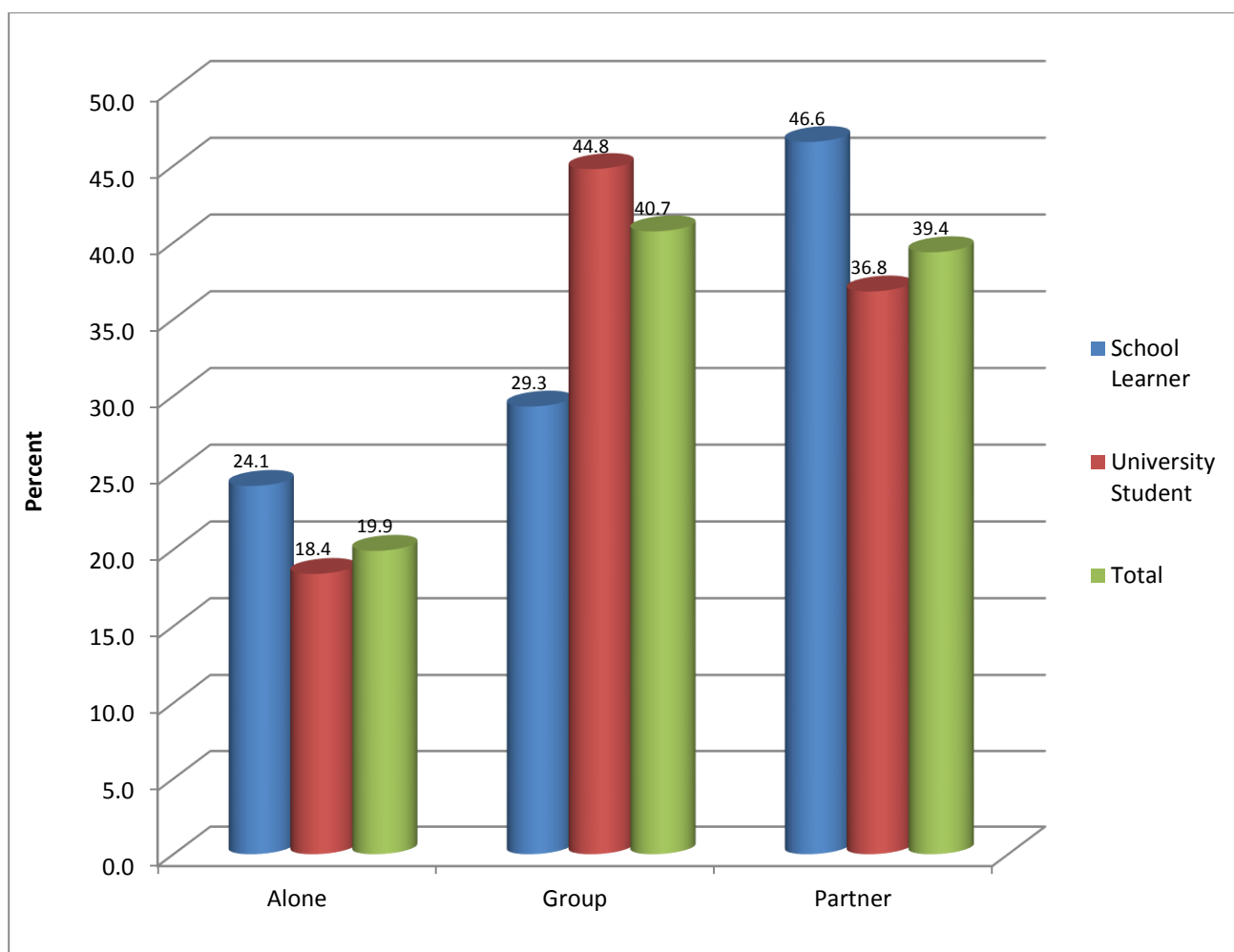


Figure 5.1: Percentages of respondents who preferred working alone, in a group or with a partner.

Overall, only about a fifth (19.9%) of respondents preferred working alone, with more school learners who preferred to work alone (24.1%) than university students (18.4%). Overall, similar numbers of respondents indicated that they would be comfortable with working in a group or with a partner. More university students preferred working in a group, whilst more school learners preferred working with a partner. Table 5.7 indicates some of the most popular reasons given for working with a partner.

Table 5.7: Some of the most popular reasons given for working with a partner

	School learners	University students	Total
Different views provided by others	27.3	37.7	35.0
Mistakes are easy to detect	9.1	10.7	10.3
Improves understanding and communication	9.1	8.2	8.4
Share knowledge and problem-solving skills	9.1	14.5	13.1

Post-test likeability factor

Figure 5.2 indicates percentage of post-test responses to the question “Do you like to solve programming problems by working alone or within a group or with a partner?”

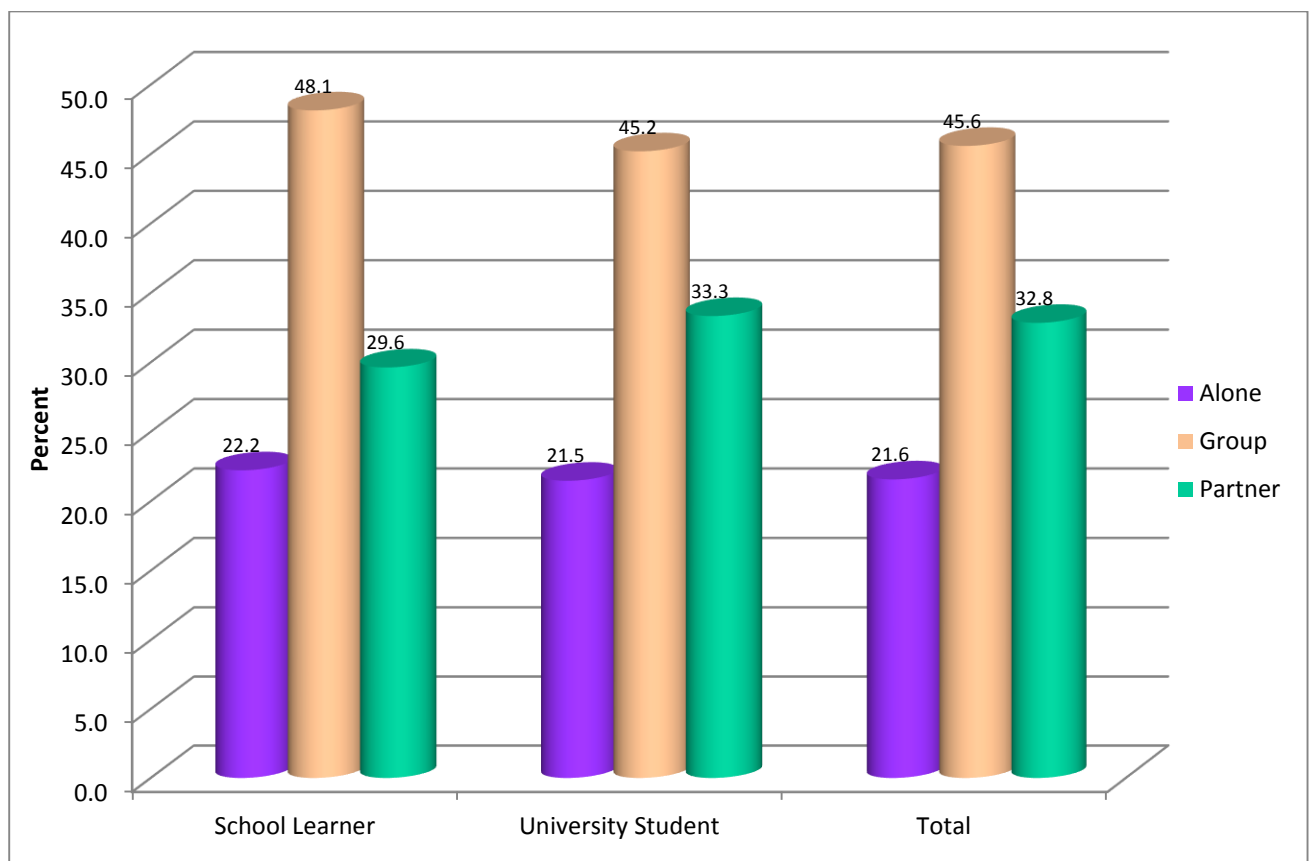


Figure 5.2: Post-test responses – ‘Do you like to solve programming problems by working alone or within a group or with a partner?’

Only about one-fifth (21.6%) of respondents preferred working alone overall.

The most popular reasons given for working with a partner were that they could

“Share different views and ideas, and that mistakes were easy to detect”.

One student summarised his experiences with group work versus PP in this rather apt quotation:

“Easier to work with partner, groups can become chaotic”.

5.5.5.4 “Enjoy ability” of programming

Responses from the pre-test as to whether the respondents enjoyed programming indie that nearly 9 out of 10 (89.1%) did so. Reasons given for this enjoyment indicated that 35.1% found programming fun and that it challenged their thinking, while 14.2% said that it created new ideas and skills. Table 5.8 indicates the level of agreement for respondents enjoying programming with a partner in the post-test.

Table 5.8: Respondents who indicated that they enjoyed programming with a partner in the post-test (%)

	Group		Total
	School learners	University students	
Yes	84.6	87.1	86.8
No	15.4	12.9	13.2

The figure indicates that there were similar levels of agreement between the learners and the students. Overall, the level of agreement with the statement was high. Some of the popular reasons given for enjoying programming with a partner are indicated in Table 5.9.

Table 5.9: Popular reasons given for enjoying PP (%)

	Total
Assist each other	24.7
Share knowledge	16.5
Different opinions are helpful	7.7
Learning experience is easier	7.1
Share work, which makes it easier to find a solution	7.1

Other reasons that were also cited were that they enjoyed hearing different styles of coding programs, enjoyed teamwork, learned from each other, and found that it improved understanding and was fun and enjoyable. This is in agreement with the literature reviewed in Chapter Two, section 2.6.3.

The two predominant reasons involved assisting each other (24.7%) and sharing of knowledge (16.5%). It was observed that because of the fact that students enjoyed programming, they were prepared to spend more time on a programming task and this eventually led to their obtaining a solution to it.

5.6 CONCLUSION

The objective of this study was to investigate the use of PP to support teaching and learning of an introductory programming course, and to provide learners and educators at schools and tertiary institutions with a strategy that they could adopt when teaching and learning introductory programming courses. The findings which emanated from the study are now summarised according to each of the research questions.

Research question one: What are learners' experiences of solving programming tasks?

The pre-test and post-test questionnaires were used to determine learners' attitudes towards programming. Learners were asked in the questionnaire whether they perceived PP to be useful to them, and responses showed strong agreement. The overall perception from the questionnaire was that learners felt that PP was a useful learning technique to assist with programming concepts. The majority of learners perceived that the use of PP improved their programming abilities. The data showed that learners felt that PP was a positive strategy and would unreservedly embrace its use.

The positive responses to the use of PP and data from the pre- and post-test suggest that overall learners and educators were confident to implement the concepts of PP.

Responses to the questionnaires and interviews showed that overall PP was easy to implement and likely to be more widely adopted by learners and educators. A majority of the learners indicated the PP was one the strategies that they would adopt when confronted by difficulties in programming tasks. The learners' attitudes towards the paired partner positively and significantly affected their intention to use PP strategies. The study revealed that use of PP is not a current practice at secondary school level, or in introductory programming courses at university.

The findings recommended that educators could implement PP and that it would enhance learning of programming concepts. This confirmed that PP would be suitable to support learning. The post-test

elicited data from learners about how programming with a partner contributed to their learning. Findings suggested that learners associated higher levels of enjoyment with programming with a partner than with programming in solitary. Observational data suggested that the male learners preferred and were more likely to select and program with an attractive female programmer. Post test data suggested that learners were extremely confident that availability of resources, paired partners, educator support enhanced their learning of programming concepts. Interview data from the educators complemented these findings. It was noted from educators that educator resources, knowledge of implementing PP, and class sizes enhanced the educators teaching experience.

This study analysed issues surrounding the adoption of a collaborative learning tool when teaching and learning programming, and tested its adoption amongst IT learners. The nine considerations of threshold concepts were used to test impact on use of the PP. The attitude of the learners would later affect his/her intention to use PP. From the results it could be said that the abovementioned constructs influenced users' attitudes towards PP. Most respondents showed a positive attitude towards the use of the PP strategy, and considered it to be a viable innovation and instructional technique for the programming classroom.

Research question two: What teaching strategies are being used to teach OO programming?

From the observations it was evident that in all cases educators used the 'chalk and talk' methodology for the teaching of OO programming. There was no evidence of the use of PP or any other collaborative teaching and learning strategy.

From the interviews it was evident that there are educators that do have an informal knowledge of PP, and in some cases this was applied in the form of group work. After the intervention (PP) it was noted that educators with a high teaching/lecture load favoured the advantage of 'less consultation time' and more self-study by pairs associated with PP. This was extracted from interview question one, where educators expressed contentment that pairs could work independently, thus freeing up teaching time.

The purpose of question two on the interview schedule was to determine the number of students per class and determine the effectiveness of PP techniques in classes with a large number of students. Classes with a relatively smaller number of programmers (less than 25) proved more conducive to applying PP, while larger classes were noisy and tended to become unmanageable.

The purpose of interview question three was to understand whether the educators had specific knowledge related to the programming language they taught, while question four related to the support that educators received. Surprisingly, educators were open and honest in conveying that the changing syllabi of IT very often meant they had to pick up a textbook and teach oneself before

teaching the class. This also raised questions for the researcher about most of the educational institutions' management approaches to training. An opportunity exists for management of schools and tertiary institutions to implement 'ongoing training workshops' to cater for this need. It must be noted that educators expressed their appreciation for learning formal methods of implementing PP. Interview question five was a direct enquiry to extract from educators specifically how PP improved their teaching.

As per the literature review (Chapter Two, section 2.6.5), educators indicated implementing PP freed up more time for them to catch up with administration work. When pairing students from similar language backgrounds, educators indicated that students tended to speak to each other in their vernacular. This was also noted in the observation schedule.

Interview questions six and seven were coupled together to extract from educators their level of competence in using PP and the intermediate results that they obtained. PP was seen as a technique to add to the educators' arsenal of strategies.

It has been found that when an educator does not have the necessary skills to teach a particular concept, they tend to resist using the technique. As mentioned earlier, educators were divided as to the implementation and use of a PP teaching strategy. Considering that this was their first formal experience of the using a PP teaching strategy, the results are encouraging.

Research question three: How does PP enhance problem solving in OO programming?

One of the findings was that the learners enjoyed discussing problem strategies with their paired partner instead of with their educators. Observations and data from the post-test showed evidence of discussions that took place between the learners in a PP environment. Learners mentioned that they consulted with the educator only after they experienced difficulties that they could not solve as a pair. Often they were able to solve the problem amongst themselves. This is reassuring, because some learners felt afraid of asking questions in class but are more comfortable communicating to a paired partner.

The following benefits were derived from the learners' interaction with PP: improved communication (collaboration); interactive participation amongst peers; enhanced accessibility of resources; a supportive and non-favourable setting; and enhancement of collaborative learning. The advantage of the pairing is that the learner can discuss their experiences with their paired partner before consulting their educator, enhancing independent learning and problem solving. Learners found it easy to identify with a paired partner from a similar linguistic group because of similarity in cultural

background, which also resulted in improved problem solving. Participating in the PP scenario also allowed for interactive participation. Learners were actively involved in the PP session, whether it was correcting their paired partner's mistake, accessing a secondary resource, or simply quietly observing the programming skills of an experienced partner. Since PP allowed learners to switch roles, it enhanced the collaboration between paired partners.

The findings of this study showed that the learners found PP an easy, efficient and enjoyable way to learn problem-solving techniques.

This chapter presented a detailed analysis of the data and a discussion of the findings gathered from the various sources of data collection. The research questions were then answered.

CHAPTER SIX

SUMMARY, RECOMMENDATIONS AND CONCLUSION

6.1 INTRODUCTION

This chapter concludes the study and presents the recommendations and a summary based on the findings. The conclusions are based on the research findings on the learners' and educators' experiences of using PP.

6.2 SUMMARY

The objective of the study was to investigate the use of PP to support teaching and learning of an introductory programming course. The study aimed to answer the following research questions: a) What are learners' experiences of solving programming tasks? b) What teaching strategies are being used to teach OO programming? and c) How does PP enhance problem solving in OO programming?

As stated in Chapter Five, it was evident that learners and educators were very positive about their experiences with PP – 86.8% of learners enjoyed programming with a partner. With the exception of one educator it was stated that PP did indeed help learners to better learn programming concepts. The positive and favourable outcomes of this study will make IT educators more interested in adopting this teaching strategy and more willing to cope with the transition from solitary programming to cooperative PP. Since much research on PP has been done in the corporate environment, the findings of this research augur well for implementation of PP in teaching and learning at tertiary and secondary school level.

Programming paradigms and languages are evolving rapidly, and becoming a more interactive part of the IT classroom. As technology improves, programming languages and paradigms will continue to evolve. The educational importance of this study is focused on cooperative learning. The results of this study will help educators understand learner perceptions regarding adoption of PP in an IT classroom or lecture room. The methodology employed in this study demonstrated a valid and reliable method for evaluating the adoption and diffusion of PP as a teaching strategy in the classroom. The results contribute to the growing body of literature related to the teaching and learning of programming in an IT classroom/lecture room. Learners' experiences of the use of PP serve as a crucial window into successful teaching strategies in programming, and served as a vital focus of the study.

Based on the educators' teaching methodologies and their experiences when using PP in the classroom, data analysis showed that the learners and educators had informal knowledge of how to use PP in IT and of its potential and challenges in teaching and learning. The research findings revealed that formal introduction of the PP strategy not only supported teaching and learning but also enhanced the teaching and learning experience of this selected group of learners. PP provided access to collaborative learning strategies, encouraged independent student learning, and created an environment for learner diversity and increased interaction and peer learning amongst the learners.

The educators indicated that the only minor setbacks in implementing PP were:

- the large numbers of students in IT classrooms at tertiary level;
- the assessment of pair programmers; and
- support of management/the Department of Education.

To be successful in PP, educators require knowledge of both PP concepts and the methodology of implementing PP. The findings of this study, which distinguish between the benefits and shortcomings of PP, indicate the need for a framework/strategy to support educators and novice programmers.

The research findings indicated that overall the learners and educators had positive experiences regarding the use of PP to support teaching and learning. The numerous observation sessions revealed that learners enjoyed working in pairs, and the pair programmers' friendship continued outside of the confines of IT programming. This was an indication that the learners looked forward to and enjoyed using PP. The data showed that the PP strategy is a beneficial and supportive technique for teaching and learning.

To review, this study has ascertained that implementing PP achieves the following

- PP contributes towards motivating students to complete a programming task or even learn a new programming concept, and the probability of successful compilation and completion is greatly increased by working in pairs.
- Paired programmers can develop strong relationships of friendship that goes beyond the programming task assigned to them, and the continuous discussion of a programming task or concept makes for more sociable and better programmers.
- The process of PP encouraged the academically weaker student to develop his programming ability by acquiring programming skills from a peer in an informal setting. Similarly, the

academically strong student acquired the humility of learning to keep his ego in check and learnt alternate programming solutions to a programming task.

- PP can afford the educator more time and create a more conducive, stress-free learning environment, whereas previously the educator was inundated with programming queries. With PP the query is discussed within the pair and most often solved within the paired partnership.
- Programming ‘mistakes’ or bugs are discussed within a pair and not in front of the entire class. Pair partners look out for each other and vehemently support their program when confronted by detractors.
- All learning and teaching environments are not conducive to implementing PP; for effective PP experiences educators need to rearrange the classroom and computer facilities.
- Educators must support the use of PP in the classroom, without such support PP is doomed from its inception.
- Sometimes it may be necessary to halt PP implementation and allow for individual programming.

Sometimes, paired programmers don't necessarily share their programming expertise with each other due to competition; however, if the educator removes the extrinsic motivation of awarding marks, then there is a greater likelihood of successfully implementing PP.

6.3 RECOMMENDATIONS FOR TEACHING AND LEARNING

Further research must be conducted by academics to investigate both learner and educator experiences when using PP in the classroom. This may also require a search for appropriate assessment tools and criteria to facilitate the testing and examination of learners exposed to PP. Development of an assessment criterion to test the effectiveness of PP activities of IT programming learners should further support the IT educators in their teaching strategies.

Furthermore, in order to achieve the benefits of PP it is recommended that secondary schools and tertiary institutions become more actively involved in collaborative and PP techniques. As an initial point of departure, educators require explicit knowledge of implementing PP. This could be as simple as a hand-out depicting the procedures to adopt when implementing PP, or a workshop with educators who actively engage in PP activities. Students must be introduced to implementing PP from their first practical lesson when assigned a programming task.

In summation, this study recommends that:

- Educators MUST ‘buy’ into the idea of implementing PP.

- Educators should remove the awarding of marks when implementing PP and concentrate on the learners' acquisition of programming concepts.
- Educators should ensure that learners are paired according to some predetermined criteria and not randomly, e.g. by gender or other cultural factors.
- Tasks should be carefully assigned to pairs and discussion should follow to ensure that all learners benefit from PP. An educator must allow for constructive criticism amongst pairs and paired groups.
- An educator must guard against paired programmers continuing without supervision. The educator must ensure control amongst learners during the programming session, and terminate the session, or decide on a swap or solitary programming if necessary.
- Educators must rearrange the seating of classrooms and laboratories so that pair programmers can conveniently programme without disturbing other programming pairs, but still have the ease of open discussion amongst paired programmers and the educator.

6.4 CONCLUSION

Previously most knowledge was gained through the 'all knowledgeable' educator who 'spoon-fed' an ignorant learner. Today the incredible power of the Internet coupled with advancements in handheld PC tablets is creating challenges to educators and learners alike. Information is virtually at their fingertips and IT educators must find novel teaching strategies to keep the programming students of today interested and eager to solve programming tasks. One such strategy is PP.

The researcher believes that if the recommendations from this study can be put into place, PP has a place in the IT classroom. PP provides an innovative way to complement the traditional learner-educator interaction. The use of PP will create opportunities for collaboration and independent learning, and meet the needs of all learners in various stages of their learning. PP provides a collaborative learning environment and a teaching strategy to cater for the programming needs of IT learners and educators of today, encompassing a multi-faceted (holistic) view of learning.

Introduction of the use of PP in our schools and tertiary institutions to support teaching and learning of programming will not only create new possibilities for our learners to engage in new ways of learning, but also provide them with job-related software development skills currently being used in the software development industry. At a simplistic level it may even provide educators with another teaching strategy that can be adopted.

This study has exposed PP concepts that have changed the researcher's learning and teaching of IT programming concepts, and expanded teaching and learning opportunities and access to job-related software strategies and resources.

The motivation for this study was to improve the learners' understanding of programming concepts and ultimately to improve the pass rate in programming subjects (Chapters One and Two).

The following points represent some key areas for potential future study:

- A study on implementing PP amongst learners from different grades and year groups, e.g. pair a student from Grade 11 with one from Grade 12.
- Pilot studies in assessment of PP and the awarding of marks and promotion based on paired assessment.
- Investigating variations and differences in how software companies implement PP, and the impact on implementing PP in academia.

It is evident that PP is a useful and a supportive strategy in teaching and learning programming at a secondary and tertiary education level.

REFERENCES

- Anderson, A, Beattie, R, & Beck, K (1998). Chrysler Goes to Extremes. *Distributed Computing*, 24-28.
- Bain, K. (2004). *What the Best College Teachers Do*, Cambridge. Cambridge,MA: Harvard University Press.
- Barker, L J, Garvin-Doxas, K, & Roberts, E. (2005). *What Can Computer Science Learn from a Fine Arts Approach to Teaching?* Paper presented at the SIGCSE St. Louis, Missouri, USA.
- Bauer, Martin W, & Gaskell, G. (2000). *Qualitative researching with text, image and sound: A practical handbook for social research*: Sage.
- Beck, K. (1999). *Extreme Programming Explained: Embrace Change*, Reading, PA: : Addison-Wesley.
- Beck, Kent. (1999). *Extreme Programming*. Paper presented at the TOOLS (29).
- Begel, A & Nagappan, N. (2008). *Pair programming: what's in it for me?* Paper presented at the Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement.
- Bell, J. (2005). *Doing Your Research Project: A Guide for First-Time Researchers* (4th edn). Maidenhead: Open University Press/McGraw-Hill.
- Berenson, Sarah B, Slaten, Kelli M, Williams, Laurie, & Ho, C. (2004). Voices of women in a software engineering course: reflections on collaboration. *Journal on Educational Resources in Computing (JERIC)*, 4(1), 3.
- Berglund, Anders, Box, Ilona, Eckerdal, Anna, Lister, Raymond, & Pears, A. (2008). *Learning educational research methods through collaborative research: the PhICER initiative*. Paper presented at the Proceedings of the tenth conference on Australasian computing education- Volume 78.
- Boeree, G C. (2008). *Personality Theories: An Introduction*. . Retrieved April 12, 2008, from <http://webspace.ship.edu/cgboer/prsonalityintroduction.html>
- Boustedt , J, Eckerdal , A, McCartney , R, Mostrom , J E, Ratcliffe , M, Sanders , K, & Zander , C. (2007). *Threshold Concepts in Computer Science:Do they exist and are they useful?* Paper presented at the ACM SIGCSE - Special Interest Group on Computer Science Education, Covington, Kentucky, USA.
- Boychuk, Duchscher, Judy, E, & Morgan, D. (2004). Grounded theory: reflections on the emergence vs. forcing debate. *Journal of Advanced Nursing*, 48(6), 605-612.
- Brought, Grant, Eby, L Martin, & Wahls, T. (2008). *The effects of pair-programming on individual programming skill*. Paper presented at the ACM SIGCSE Bulletin.

- Breed, Betty, Mentz, Elsa, Havenga, Marietjie, Govender, Irene, Govender, Desmond, Dignum, Frank, & Dignum, V. (2013). Views of the Use of Self-directed Metacognitive Questioning during Pair Programming in Economically Deprived Rural Schools. *African Journal of Research in Mathematics, Science and Technology Education*, 17(3), 206-219.
- Bryant, Sallyann, Romero, Pablo, & du Boulay, B. (2006). The collaborative nature of pair programming *Extreme programming and agile processes in software engineering* (pp. 53-64): Springer.
- Carter, L. (2006). Why students with an apparent aptitude for computer science don't choose to major in computer science. *ACM SIGCSE Inroads Bulletin*, 38(1), 27-31.
- Charmaz, Kathy. (2003). Grounded theory. *Strategies of qualitative inquiry*, 2, 249.
- Chong, Jan, & Hurlbutt, T. (2007). *The social dynamics of pair programming*. Paper presented at the Software Engineering, 2007. ICSE 2007. 29th International Conference on.
- Cliburn, Daniel C. (2003). Experiences with pair programming at a small college. *Journal of Computing Sciences in Colleges*, 19(1), 20-29.
- Cockburn, Alistair, & Williams, L. (2001). The Costs and Benefits of Pair Programming, in *Extreme Programming Examined*.
- Cohen, L, Manion, L, & Morrison, K. (2000). *Research Methods in Education* (5th ed.). New York: Routledge Falmer.
- Cooper, Stephen, Dann, Wanda, & Pausch, R. (2003). *Teaching objects-first in introductory computer science*. Paper presented at the ACM SIGCSE Bulletin.
- Cousin, Glynis, & Meyer, J. (2006). Threshold concepts, troublesome knowledge and emotional capital. *Overcoming barriers to student understanding: Threshold concepts and troublesome knowledge*, 134-147.
- Creswell, John W. (2013). *Qualitative inquiry and research design : choosing among five approaches* (3rd ed.): Thousand Oaks : SAGE Publications.
- Davidson, Neil. (1994). Cooperative and collaborative learning: An integrative perspective. *Creativity and collaborative learning: A practical guide to empowering students and teachers*, 13-30.
- Davis, Barbara Gross. (2009). *Tools for teaching*: Wiley. com.
- De Vos, AS, Strydom, Herman, Fouche, CB, & Delport, C. (2002). Research at grassroots: For the social sciences and human service profession. *Pretoria: Van Schaik Publishers*.
- DeClue, Timothy H. (2003). Pair programming and pair trading: effects on learning and motivation in a CS2 course. *Journal of Computing Sciences in Colleges*, 18(5), 49-56.
- Denzin, Norman K, & Lincoln, Y. (2000). The discipline and practice of qualitative research. *Handbook of qualitative research*, 2, 1-28.
- Du Plooy, Gertruida M. (2009). *Communication research: techniques, methods and applications*: Juta and Company Ltd.

- Duffy, Mary. (1987). Methodological triangulation: a vehicle for merging quantitative and qualitative research methods. *Journal of Nursing Scholarship*, 19(3), 130-133.
- Education, Department of. (2003). Draft White Paper on e-Education: Transforming Learning and Teaching through Information and Communication Technologies (ICTs). *Government Gazette*(26734), 1-44.
- Efklides, Anastasia. (2006). Metacognition and affect: What can metacognitive experiences tell us about the learning process? *Educational Research Review*, 1(1), 3-14.
- Ellsworth, Elizabeth. (1997). *Teaching positions: Difference, pedagogy, and the power of address*: ERIC.
- Field, R. M. (2005). *Favourable conditions for effective and efficient learning in a blended face-to-face/online method*. Ascilite.
- Fowler, Martin. (1999). *Refactoring: improving the design of existing code*: Addison-Wesley Professional.
- Galorath, Daniel D, & Evans, M W. (2006). *Software sizing, estimation, and risk management: when performance is measured performance improves*: CRC Press.
- Glaser, & Strauss, AL. (1967). *The discovery of Grounded Theory Strategies for Qualitative Research* London: Weidenfield and Nicolson.
- Glaser, Barney G. (2001). *The grounded theory perspective: Conceptualization contrasted with description*: Sociology Press.
- Glaser, Barney G, & Strauss, A. (2009). *The discovery of grounded theory: Strategies for qualitative research*: Transaction Books.
- Govender, I. (2006). *Learning to program, learning to teach programming: pre-and-in-service teachers' experiences of an object-oriented language*. (PHD), Unisa, Pretoria.
- Hanks, B, McDowell, C, D, Draper, & Krnjajic, M. (2004, 28-30 June). *Program quality with pair programming in CS1*. Paper presented at the 9th Annual Conference on Innovation and Technology in Computer Science Education.
- Hannay, Jo E, Dybå, Tore, Arisholm, Erik, & Sjøberg, D. (2009). The effectiveness of pair programming: A meta-analysis. *Information and Software Technology*, 51(7), 1110-1122.
- Hatch, J Amos. (2002). *Doing qualitative research in education settings*: SUNY Press.
- Havenga, M, De Villiers, R, & Mentz, E. (2011). Thinking processes used by high-performing students in a computer programming task.
- Havenga, M, & Mentz, E. (2009). *The school subject Information Technology: A South African perspective*. Paper presented at the South African Computer Lecturers Association Conference (SACLA), Mpekweni Beach Resort, South Africa.
- Havenga, M, Mentz, Elsa, & de Villiers, R. (2008). Knowledge, skills and strategies for successful object Knowledge, skills and strategies for successful object-oriented p oriented

- programming: a proposed learning repertoire programming: a proposed learning repertoire
 programming: a proposed learning repertoire. *South African Computer Journal*, 4, 1.
- Heather, Nick, Rollnick, Stephen, & Bell, A. (1993). Predictive validity of the Readiness to Change Questionnaire. *Addiction*, 88(12), 1667-1677.
- Henning, Elizabeth. (2004). Crossing the digital divide safely and trustingly: how ecologies of learning scaffold the journey. *Computers and education*, 42(4), 333-352. doi: 10.1016/j.compedu.2003.08.006
- Henning, Elizabeth, & Van der Westhuizen, D. (2004). Crossing the digital divide safely and trustingly: how ecologies of learning scaffold the journey. *Computers & Education*, 42(4), 333-352.
- Ho, Chih-Wei, Raha, Somik, Gehringer, Edward, & Williams, L. (2004). *Sangam: a distributed pair programming plug-in for Eclipse*. Paper presented at the Proceedings of the 2004 OOPSLA workshop on eclipse technology eXchange.
- Horn, Elizabeth M, Collier, William G, Oxford, Julie A, Bond Jr, Charles F, & Dansereau, D. (1998). Individual differences in dyadic cooperative learning. *Journal of Educational Psychology*, 90(1), 153.
- Howard, Elizabeth V. (2006). Attitudes on using pair-programming. *Journal of Educational Technology Systems*, 35(1), 89-103.
- Howe, Emily, Thornton, Matthew, & Weide, Bruce W. (2004). *Components-first approaches to CS1/CS2: principles and practice*. Paper presented at the ACM SIGCSE Bulletin.
- Howles, T. (2009). A study of the attrition and the use of student learning communities in the computer science introductory programming sequence. *Computer science education*, 19(1), 1-13.
- Hussey, Trevor, & Smith, P. (2003). The Uses of Learning Outcomes. *Teaching in Higher Education*, 8(3), 357-368. doi: 10.1080/13562510309399
- Ismail , M N, Ngah , N A, & Umar , I N. (2010). Instructional Strategy in the teaching of computer programming:a need assessment analyses. *TOJET: The Turkish Online Journal of Educational Technology*, 9(2).
- Ismail, Mohd Nasir, Azilah, NGAH, Naufal, UMAR, & Kelantan, Universiti Teknologi Mara Cawangan. (2010). Instructional strategy in the teaching of computer programming: a need assessment analyses. *TOJET*, 9(2), 125-131.
- Jandaghi, Gholamreza, & Shaterian. (2008). Validity, reliability, and difficulty indices for instructor-built exam questions. *Journal of Applied Quantitative Methods*, 3(2), 151-155.
- Johnson, D W, & Johnson, R T. (1992). *Positive Independence. Key to effective cooperation* New York: Cambridge University Press.
- Johnson , D W, & Johnson , R T. (2009). An educational psychology success story: Social interdependence theory and cooperative learning. *Educational researcher*, 38(5), 365-379.

- Johnson , David W, Johnson , Roger T, & Smith , Karl. (2007). The state of cooperative learning in postsecondary and professional settings. *Educational Psychology Review*, 19(1), 15-29.
- Johnson, R, & Waterfield, J. (2004). Making words count: the value of qualitative research. *Physiotherapy Research International*, 9(3), 121-131.
- Kinnunen, P, & Malmi, L. (2006). *Why students drop out CS1 course?* . Paper presented at the International Workshop on Computing Education Research, New York.
- Klopper, Hester. (2008). The qualitative research proposal. *Curationis*, 31(4), 62-72.
- Kolling, M, & Rosenberg, J. (2001). *Guidelines for Teaching Object Orientation with Java*. Paper presented at the 6th conference on Information Technology in Computer Science Education, Canterbury.
- Koorsse, M, Calitz, A P, & Cilliers, C C. (2010). *Programming in South African Schools: The Inside Story*. Paper presented at the South African Computer Lecturers Association (SACLA), University of Pretoria.
- Krefting, Laura. (1991). Rigor in qualitative research: The assessment of trustworthiness. *The American journal of occupational therapy*, 45(3), 214-222.
- Land, R , Cousin, Glynis, Meyer, Jan HF, & Davies, Peter. (2005). Threshold concepts and troublesome knowledge (3): implications for course design and evaluation. *Improving Student Learning–equality and diversity*, Oxford: OCSLD.
- Land, R, Cousin, G, Meyer, J H F, & Davies, P. (2005). *Threshold concepts and troublesome knowledge implications for course design and evaluation*, in: 53–64. Oxford, OCSLD.
- Lang , J, & Ottinger , T. (2011). Pair Programming Benefits Two Heads Are Better than One. from <http://pragprog.com/magazines/2011-07/pair-programming-benefits>
- Lather, Patti. (1998). Critical pedagogy and its complicities: A praxis of stuck places. *Educational theory*, 48(4), 487-497.
- Leedy, Paul D, & Ormrod, J. (2001). *Practical research*: Pearson education international.
- Liebenberg J. (2010). *Secondary school girls' experiences of pair programming in information technology*. (MEd), Potchefstroom: NWU.
- Liebenberg, Janet, Mentz, Elsa, & Breed, B. (2012). Pair programming and secondary school girls' enjoyment of programming and the subject Information Technology (IT). *Computer Science Education*, 22(3), 219-236.
- Likert, Rensis. (1932). A technique for the measurement of attitudes. *Archives of psychology*.
- Macgregor, K. (2007b). South Africa: Student drop-out rates alarming. 0003. 2010, from <http://www.universityworldnews.com/article.php?story=20071025102245380&mode=print>
[Accessed: 2009-05-29]
- Marais , H C, & Mouton , Johann. (1988). *Basic concepts: The methodology of the social sciences*: HSRC Press.

- McDowell , C, Werner , L, Bullock , H , & Fernald , L. (2002). *The Effects of Pair-Programming on Performance in an Introductory Programming Course*. Paper presented at the SIGCSE'02 . Covington, Kentucky, USA.
- McKinney , D, & Denton , L.F. (2004). Houston, we have a problem: There's a leak in the CS1 affective oxygen tank. *ACM SIGCSE Inroads Bulletin*, 36(1), 236-239.
- McMillan , J H, & Schumacher , S. (1993). *Research in Education. A Conceptual Introduction* (third ed.). New York: Harper Collins College Publishers.
- McWhaw, K, Schnackenberg, H, Sclate , R & Abrami , P C. (2003). From co-operation to collaboration: Helping students become collaborative learners.Co-operative Learning: The social and intellectual outcomes of learning in groups. .
- Mendes , Emilia, Al-Fakhri , Lubna Basil, & Luxton-Reilly , Andrew. (2005). *Investigating pair-programming in a 2 nd-year software development and design computer science course*. Paper presented at the ACM SIGCSE Bulletin.
- Mentz , E, & Goosen , L. (2007). Are groups working in the information technology class? . *South African Journal of Education*, 27(2), 329-343.
- Mentz , E, Van der Walt , J L, & Goosen, L. (2008). The effect of incorporating cooperative learning principles in pair programming for student teachers. *Computer Science Education*, 18(4), 247-260.
- Mentz, E, Van Der Walt, J.L, & Goosen, L. (2008). The effect of incorporating cooperative learning principles in pair programming for student teachers. *Computer science education*, 18(4), 247-260.
- Meyer , J H , & Land , R. (2005). Threshold concepts and troublesome knowledge Epistemological considerations and a conceptual framework for teaching and learning. *Higher Education*, 49, 373-388.
- Meyer, J, & Land, R. (2003). *Threshold concepts and troublesome knowledge: linkages to ways of thinking and practising within the disciplines*: University of Edinburgh.
- Molokken, Kjetil, & Jorgensen, M. (2003). *A review of software surveys on software effort estimation*. Paper presented at the Empirical Software Engineering, 2003. ISESE 2003. Proceedings. 2003 International Symposium on.
- Nagappan, Nachiappan, Williams, Laurie, Ferzli, Miriam, Wiebe, Eric, Yang, Kai, Miller, Carol, & Balik, S. (2003). *Improving the CS1 experience with pair programming*. Paper presented at the ACM SIGCSE Bulletin.
- Newman, Daniel S. (2012). A Grounded Theory Study of Supervision of Preservice Consultation Training. *Journal of Educational and Psychological Consultation*, 22(4), 247-279.
- Nosek , T. (1998). The Case for Collaborative Programming. *Communications of the ACM*, 105-108.
- Okur , M C. (2006). Teaching Object Oriented Programming at the Introductory Level. *Journal of Yasar University*, 1(2), 149-157.

- Padmanabhuni , Venkata Vinod Kumar, Tadiparthi , Hari Praveen, Yanamadala , Muralidhar, & Madina , S. (2012). Effective pair programming practice-an experimental study. *Journal of Emerging Trends in Computing and Information Sciences*, 3(4), 471-479.
- Perkins , David, Meyer , JHF, & Land , R. (2006). Constructivism and troublesome knowledge. *Overcoming barriers to student understanding: Threshold concepts and troublesome knowledge*, 33-47.
- Pikkarainen , Minna, Haikara , Jukka, Salo , Outi, Abrahamsson , Pekka, & Still , J. (2008). The impact of agile practices on communication in software development. *Empirical Software Engineering*, 13(3), 303-337.
- Preston. (2005). Pair programming as a model of collaborative learning: a review of the research. *The Journal of Computing Sciences in Colleges*, 20(4), 39.
- Preston , D. (2005). Pair programming as a model of collaborative learning: a review of the research. *Journal of Computing Sciences in colleges*, 20(4), 39-45.
- Preston , D. (2006). Using Collaborative Learning Research to Enhance Pair Programming Pedagogy. *ACM SIGITE Newsletter*, 3(1).
- Ramsden , P. (1992). *Learning to teach in higher education*. London: Routledge.
- Rountree, J & Rountree, N. (2009). *Issues regarding threshold concepts in computer science*. Paper presented at the Proceedings of the Eleventh Australasian Conference on Computing Education-Volume 95.
- Roy , Peter, & Haridi . (2003). Teaching Programming Broadly and Deeply: The Kernel Language Approach. In L. Cassel & R. Reis (Eds.), *Informatics Curricula and Teaching Methods* (Vol. 117, pp. 53-62): Springer US.
- Salleh , Norsaremah. (2008). *A Systematic Review of Pair Programming Research-Initial Results*. Paper presented at the Proc. New Zealand Computer Science Research Student Conference (NZCSRSC08), Christchurch.
- Salleh , Norsaremah, Mendes , Emilia, Grundy , John, & Burch. (2010). *An empirical study of the effects of conscientiousness in pair programming using the five-factor personality model*. Paper presented at the Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1.
- Seidman , Irving. (1998). *Interviewing as Qualitative Research: A Guide for Researchers in Education and the Social Sciences*, by Irving Seidman (2nd ed.). New York: Teachers College Press.
- Shanahan , Martin, & Meyer . (2006). The troublesome nature of a threshold concept in economics. *Overcoming barriers to student understanding: Threshold concepts and troublesome knowledge*, 100-114.

- South Africa. (2012). *The Green Paper for Post School Education and Training*. South Africa:Retrieved from <http://www.dhet.gov.za/LinkClick.aspx?fileticket=w0qJyEiFVYQ%3d&tabid=189&mid=483>
- Spicer , N. (2004). *Combining qualitative and quantitative methods. Researching Society and Culture* (C. Searle Ed.). New Delhi: Sage Publications.
- Strauss , A, & Corbin , J. (1998). Basics of Qualitative Research. 1998. *Thousand Oaks*.
- Swamidurai , Rajendran, & Umphress, D. (2012). Collaborative-Adversarial Pair Programming. *ISRN Software Engineering*, 2012, 11. doi: 10.5402/2012/516184
- Terre Blanche , M, & Durrheim , K (2002). *Research in Practice*. Cape Town: University of Cape Town Press.
- Thatcher , A. (2007). Using the World Wide Web to support classroom lectures in a psychology course. *South African Journal of Psychology*,, 37(2), 348-353.
- Tillema , J, & van der Westhuizen , G J. (2006). Knowledge construction in collaborative enquiry among teachers. *Teachers and Teaching: theory and practice*, 12(1), 51-67.
- Van Roy , P, & Haridi , S. (2002). Concepts, Techniques, and Models of Computer Programming.
- Vandegrift, T. (2004, 3-7 March 2004). *Coupling pair programming and writing : Learning about students'perceptions and processes*. . Paper presented at the 35th SIGCSE Technical Symposium on Computer Science Education
- Veenman , Simon, van Benthum , Niek, Bootsma , Dolly, van Dieren , Jildau, & van der Kemp , Nicole. (2002). Cooperative learning and teacher education. *Teaching and teacher education*, 18(1), 87-103.
- Ventura , P, & Ramamurthy , B. (2004). Wanted: CS1 students. No experience required. *ACM SIGCSE Inroads Bulletin*, 36(1), 240-244.
- Werner , Linda L, Hanks , Brian, & McDowell , C. (2004). Pair-programming helps female computer science students. *Journal on Educational Resources in Computing (JERIC)*, 4(1), 4.
- Wexelblat , Richard L. (1978). *History of programming languages I*: ACM.
- Williams , Laurie, & Kessler , R. (2003). *Pair programming illuminated*: Addison-Wesley Professional.
- Williams , Laurie, Kessler , Robert R, Cunningham , Ward, & Jeffries , R. (2000). Strengthening the case for pair programming. *Software, IEEE*, 17(4), 19-25.
- Williams , Laurie, & Upchurch , R. (2001). In support of student pair-programming. *ACM SIGCSE Bulletin*, 33(1), 327-331.
- Williams , Laurie, Wiebe , Eric, Yang , Kai, Ferzli , Miriam, & Miller. (2002). In support of pair programming in the introductory computer science course. *Computer Science Education*, 12(3), 197-212.
- Zaidman , Marsha. (2011). *Inspiring future female scientists*. Paper presented at the Frontiers in Education Conference (FIE), 2011.

APPENDIX A: PRE-TEST QUESTIONNAIRE

UNIVERSITY OF KWAZULU-NATAL
SCHOOL OF SCIENCE, MATHS & TECHNOLOGY EDUCATION

MEd (Computer Science) Research Project

Researcher: T.P Govender (031 373 5553 / 0845743807)

Supervisor: Dr DW Govender @ UKZN (031 260 3428)

UKZN Research Office: (031 260 8350)

A Study of using a cooperative learning technique when teaching object oriented programming (OOP).

The purpose of this survey is to obtain information from academics/learners regarding how secondary schools and tertiary institutions in KZN can successfully implement a cooperative learning technique when learning programming. The data that you provide will help us identify ways of using these techniques to provide teachers and learners with a richer teaching and learning experience in programming.

The primary aim of this initial questionnaire is to determine the extent and nature of teaching and learning OOP.

The questionnaire should take only 10 minutes to complete, you are asked to indicate what is true for you, so there are no “right” or “wrong” answer to any question.

If you wish to make a comment please write it down in question 11 .Please make sure that you do not skip any questions.

Thank you for participating.

PRE-TEST

THE QUESTIONNAIRE

PERSONAL INFORMATION _____

STUDENT NUMBER: _____

GENDER: _____

GRADE: _____

1 WHAT IS YOUR HIGHEST QUALIFICATION IN COMPUTER PROGRAMMING?

GRADE	GRADE	Grade	1 ST	2 ND	3 RD
10	11	12	YEAR	YEAR	YEAR

2 DO YOU HAVE ANY PREVIOUS COMPUTER PROGRAMMING EXPERIENCE

YES	NO
-----	----

3 IN WHICH PROGRAMMING LANGUAGE DO YOU CURRENTLY PROGRAM?

YOU MAY SELECT MORE THAN ONE OPTION (use a tick to indicate your choice)

JAVA /JAVASCRIPT	
DELPHI	
C++	
C#	
VISUAL BASIC	
OTHER	
IF OTHER PLEASE SPECIFY	

4 HOW WOULD YOU RATE YOURSELF AS A COMPUTER PROGRAMMER? Insert a tick (✓)

advanced		good		struggling	
----------	--	------	--	------------	--

Give a reason for your choice.

5. Do you like to solve programming problems by working alone or within a group or with a partner?

Insert a tick (✓)

Alone	Group	Partner
-------	-------	---------

Please give reasons for your choice

6, The following scale is used for the statements below:

Never: The statement would never be true of you.

Seldom: The statement would seldom be true of you.

Often: The statement would often describe you.

Always: The statement would be true of you all the time.

For each statement, tick the most appropriate column which best describes you.

	Statement	never	seldom	often	always
1	When programming I always achieve my programming solutions				
2	When I write programs on my own the programs are of a higher quality				
3	My programs usually have syntax, run time and logic errors				
4	I usually obtain a programming solution within the allocated time period				
5	I enjoy programming on my own				
6	I feel motivated to find a solution to a programming task				
7	I feel frustrated when I am unable to successfully compile my program				
8	I feel confident that on my own I can find a solution to a programming task				
9	I have the necessary problem solving skills to find a solution to a programming task				
10	I am able to quickly learn and grasp programming concepts				

7. Do you enjoy programming?

YES	NO
-----	----

Explain why or why not

8. When your teacher gives you a programming task how do you start to program the solution?

9. If you experience difficulty developing a program to solve a problem, what do you do?

10. If you have any additional comments with respect to programming please write them here

APPENDIX B: POST TEST QUESTIONNAIRE

UNIVERSITY OF KWAZULU-NATAL
SCHOOL OF SCIENCE, MATHS & TECHNOLOGY EDUCATION
MEd (Computer Science) Research Project
Researcher: T.P Govender (031 373 5553 / 0845743807)
Supervisor: Dr DW Govender @ UKZN (031 260 3428)
UKZN Research Office: (031 260 8350)

A Study of using a cooperative learning technique when teaching object oriented programming (OOP).

The purpose of this survey is to obtain information from academics/learners regarding how secondary schools and tertiary institutions in KZN can successfully implement a cooperative learning technique when learning programming. The data that you provide will help us identify ways of using these techniques to provide teachers and learners with a richer teaching and learning experience in programming.

The primary aim of this initial questionnaire is to determine the extent and nature of teaching and learning OOP.

The questionnaire should take only 10 minutes to complete, you are asked to indicate what is true for you, so there are no “right” or “wrong” answer to any question.

Please answer honestly. Your input is important to this research. All information is treated in the strictest of confidence.

If you wish to make a comment please write it down in question 11. Please make sure that you do not skip any questions.

Thank you for participating.

POST TEST

THE QUESTIONNAIRE

PERSONAL INFORMATION

STUDENT NUMBER: _____

GENDER: _____

GRADE: _____

WHAT IS YOUR HIGHEST QUALIFICATION IN COMPUTER PROGRAMMING?

GRADE 10	GRADE 11	Grade 12	1 ST YEAR	2 ND YEAR	3 RD YEAR
-------------	-------------	-------------	-------------------------	-------------------------	-------------------------

DO YOU HAVE ANY PREVIOUS COMPUTER PROGRAMMING EXPERIENCE

YES	NO
-----	----

IN WHICH PROGRAMMING LANGUAGE DO YOU CURRENTLY PROGRAM?

YOU MAY SELECT MORE THAN ONE OPTION (use a tick to indicate your choice)

JAVA /JAVASCRIPT	
DELPHI	
C++	
C#	
VISUAL BASIC	
OTHER	
IF OTHER PLEASE SPECIFY	

HOW WOULD YOU RATE YOURSELF AS A COMPUTER PROGRAMMER AFTER WORKING WITH A PARTNER? Insert a tick

advanced		good		Struggling	
----------	--	------	--	------------	--

Give a reason for your choice?

Do you like to solve programming problems by working alone or within a group or with a partner?

Alone	Group	Partner
-------	-------	---------

Please give reasons for your choice.

The following scale is used for the statements below:

Never: The statement would never be true of you.

Seldom: The statement would seldom be true of you.

Often: The statement would often describe you.

Always: the statement would be true of you all the time.

For each statement, tick the most appropriate column which best describes you.

	Statement	never	seldom	often	always
1	We achieve our programming solution				
2	When I write programs with a fellow student the programs are of a higher quality				
3	Our programs usually have syntax, run time and logic errors				
4	We usually obtain a programming solution within the allocated time period				
5	We enjoy programming in pairs				
6	Working as a pair, together we feel motivated to find a solution to a programming task				
7	Working as a pair we share each other's frustration when we are unable to successfully compile OUR program				
8	We feel confident that working together we can find a solution to a programming task				
9	We have the necessary problem solving skills to find a solution to a programming task				
10	Together we can quickly learn and grasp programming concepts				

Do you enjoy programming with a partner?

YES	NO
-----	----

Explain why or why not

When your teacher gives you a programming task to work in a pair/group how do you start to program the solution?

If you (in your paired group) experience difficulty developing a program to solve a problem, what do you and your partner do?

If you have any additional comments with respect to programming please write them here

APPENDIX C: QUESTIONNAIRE FOR EDUCATORS

statement	never	seldom	often	always
1 My students achieve their programming solutions				
2 Students programming independently produce programs of a higher quality				
3 Individual programs usually have syntax, run time and logic errors				
4 Students produce a programming solution within the allocated time period				
5 Students enjoy programming on their own				
6 Students feel motivated to find a solution to a programming task				
7 Students feel frustrated when they are am unable to successfully compile their programs				
8 Students are confident that on their own(without my help) they can find a solution to a programming task				
9 Students have the necessary problem solving skills to independently find a solution to a programming task				
10 Students are able to quickly learn and grasp programming concepts				
11 I am able to cope with the marking of tasks and my teaching work load				
12 I am able to cope with students questions and requests for assistance during and after class				
13 My students are actively engaged in programming tasks				
14 My students are motivated and really want to work on their own to find a solution to the problem				
15 My students enjoy programming				
16 I am able to offer assistance to the high flyers/average / and below average programmers				
17 Students easily give up and wait for solutions from the teacher or fellow students				
18 I experience disciplinary problems during programming classes				
19 I am able to monitor all my students progress				
20 I am able to differentiate various levels of programming tasks according to students' abilities				

APPENDIX D: OBSERVATION SCHEDULE

OBSERVATION OF TEACHERS AND LEARNERS DURING A PROGRAMMING LESSON (pre-test / post-test)			
		YES	NO
1	Is the teacher using the pair programming?		
2	Is the teacher confident in using co-operative/pair programming?		
3	Is the seating arrangement conducive to the principles of pair programming?		
4	Do the learners show evidence of pair programming?		
5	Are all learners actively involved in the lesson?		
6	Are learners well behaved and attentive during the lesson?		
7	Do learners have the opportunity to question the teacher during the lesson?		
8	Are the learners' work also shown on the smart board/White board?		
9	Is there any evidence to indicate that the use of the pair programming is facilitating the learners understanding the lesson?		
10	Does the teacher have the necessary skills/knowledge to introduce pair programming?		
11	Do learners seem to enjoy programming (pre-test/post-test)?		

APPENDIX E: LETTER OF PERMISSION

I would really appreciate it if you could spend some time to complete this questionnaire. The information you provide will be used solely for the purposes of an academic research project for a Masters of Education thesis at UKZN. You will not be identified by name in any report and all information shall be treated confidentially. Please sign to indicate that you have read this and give your informed consent to participate in this research project. You may withdraw your consent and your data at any time.

Many thanks for your co-operation

Prinavin Govender

Student no: 891288849

Dr. D.W Govender

Project supervisor

I give permission to use the data from this questionnaire for academic research purposes.

I understand that I will not be identified by name in any research report and that I am not under any obligation to give this permission. I may withdraw my permission at a later stage if I so wish.

Signed: _____ Date: _____

NAME OF SCHOOL/UNIVERSITY: _____

GENDER : _____

A QUESTIONNAIRE FOR TEACHERS

I would really appreciate it if you could spend some time to complete this questionnaire. The information you provide will be used solely for the purposes of an academic research project for a Masters of Education thesis at UKZN. You will not be identified by name in any report and all information shall be treated confidentially. Please sign to indicate that you have read this and give your informed consent to participate in this research project. You may withdraw your consent and your data at any time.

Many thanks for your co-operation

Prinavin Govender

Dr. D.W Govender

Student no: 891288849

Project supervisor

I give permission to use the data from this questionnaire for academic research purposes.

I understand that I will not be identified by name in any research report and that I am not under any obligation to give this permission. I may withdraw my permission at a later stage if I so wish.

Signed: _____ Date: _____

NAME OF SCHOOL: _____

GENDER : _____

Informed Consent Form

CONSENT TO PARTICIPANT IN RESEARCH

15 November 2011

The title of my proposed research is “A Study of using a cooperative learning technique when teaching object oriented programming”.

You are asked to participate in a research conducted by Mr. T.P Govender, a Masters student at the University of Kwa-Zulu Natal from the Department of Computer Science Education. Contact details are as follows:

Email: prinaving@dut.ac.za

Cell: 0845743807

The purpose of this study is to determine how a cooperative pair programming technique can be implemented to improve teaching and learning in an introductory object oriented programming paradigm. Further this study will examine the perceptions and experiences of Kwa-Zulu Natal teachers and learners when using pair programming to facilitate the teaching and learning process in the classroom.

If you are willing to participant in this research study, you will be exposed to a “pair programming environment” in your learning environment and be observed by the researcher. Thereafter you are required to complete a questionnaire.

This study will be conducted over a 2 month period. You have been identified as a participant in this research study because you are currently using a programming paradigm in your teaching and learning process.

As your participation in this study is voluntary, you can stop participating in the study at any time, for any reason, if you so decide. Your decision to stop participating, or to refuse to answer particular questions, will not affect your relationship with the researcher or the school.

All information you supply during the research will be held in confidence and unless you specifically indicate your consent, your name will not appear in any report or publication of the research. Your data will be safely stored in a locked facility and only research staff will have access to this

information. Confidentiality will be provided to the fullest extent possible by law. All data collected will be kept for five years, thereafter all documents will be shredded.

In your participation in this study there will be no cost incurred by you and you may not benefit personally from this study however your contributions will assist in the teaching and learning of object oriented programming.

Your contribution to this study will be kept confidentially and your anonymity will be maintained in the presentation of the findings in this study.

If you have any questions about this study, please feel free to contact:

Dr. D.W. Govender, University of Kwa-Zulu Natal-Department of Computer Science Education:
Tell: 031 2603428 Email: govenderd50@ukzn.ac.za

Declaration

I _____, (full name of participant), hereby confirm that I understand the contents of this document and the nature of the research project, and I consent to participating in the research project.

I understand that I am at liberty to withdraw from the project at any time, should I so desire.

Signature of Participant

Date

Informed Consent Form

CONSENT TO PARTICIPANT IN RESEARCH

The title of my proposed research is “A Study of using a cooperative learning technique when teaching object oriented programming”.

You are asked to participate in a research conducted by Mr. T.P Govender, a Masters student at the University of Kwa-Zulu Natal from the Department of Computer Science Education. Contact details are as follows:

Email: prinaving@dut.ac.za

Cell: 0845743807

The purpose of this study is to determine how a cooperative pair programming technique can be implemented to improve teaching and learning in an introductory object oriented programming paradigm. Further this study will examine the perceptions and experiences of Kwa-Zulu Natal teachers and learners when using pair programming to facilitate the teaching and learning process in the classroom.

If you are willing to participant in this research study, you will be exposed to a “pair programming environment” in your learning environment and be observed by the researcher. Thereafter you are required to complete a questionnaire.

This study will be conducted over a 2 month period. You have been identified as a participant in this research study because you are currently using a programming paradigm in your teaching and learning process.

As your participation in this study is voluntary, you can stop participating in the study at any time, for any reason, if you so decide. Your decision to stop participating, or to refuse to answer particular questions, will not affect your relationship with the researcher or the school.

All information you supply during the research will be held in confidence and unless you specifically indicate your consent, your name will not appear in any report or publication of the research. Your data will be safely stored in a locked facility and only research staff will have access to this

information. Confidentiality will be provided to the fullest extent possible by law. All data collected will be kept for five years, thereafter all documents will be shredded and audio tape will be erased.

In your participation in this study there will be no cost incurred by you and you may not benefit personally from this study however your contributions will assist in the teaching and learning process. Your contribution to this study will be kept confidentially and your anonymity will be maintained in the presentation of the findings in this study.

If you have any questions about this study, please feel free to contact:

Dr. D.W. Govender, University of Kwa-Zulu Natal-Department of Computer Science Education:

Tell: 031 2603428 Email: govenderd50@ukzn.ac.za

-----✂-----✂-----✂-----✂-----✂

Declaration

I _____, (full name of participant), hereby confirm that I understand the contents of this document and the nature of the research project, and I consent to participating in the research project.

I understand that I am at liberty to withdraw from the project at any time, should I so desire.

Signature of Participant

Date

Letter of consent to parent/guardian

Dear parent/guardian

My name is Prinavin Govender. I am an educator at Durban University of Technology. I am presently completing my Masters degree in Computer Science Education at the University of Kwa-Zulu Natal and Dr Desmond Govender is my supervisor. Telephone number: 031 260 3689

One of the criteria for completing my degree is to conduct a research-study on my chosen field of research.

My research study explores the perceptions and experiences of Kwa-Zulu Natal teachers' and learners' when using pair programming to facilitate the teaching and learning of a programming course in the classroom. My research title is:

"A Study of using a cooperative learning technique when teaching object oriented programming"

The research involves your child/ward to fill a questionnaire. All ethical considerations will be strictly maintained at all times. All information provided will be kept in strict confidence. Please note that participation in this research is voluntary and your child/ward may withdraw from participating at any time he/she feels the need to do so. Furthermore, your child/ward needs your consent to participate. If you consent to your child being part of this research please sign the form below.

-----✂-----✂-----✂-----✂-----✂

I, parent/guardian of ----- give consent for his/her participation in the research project.

PARENT/GUARDIAN

DATE

Thank you for your co-operation.

RESEARCHER

MR .T.P Govender

CONTACT NUMBERS: 0845743807

0312088282

70 Pastoral road
Asherville
Durban
4091

15 November 2011
Department of Education and Culture

Dear Sir/Madam

RE: APPLICATION FOR PERMISSION TO CONDUCT RESEARCH AT
.....SECONDARY SCHOOL:

I am a student at the University of Kwa-Zulu Natal. I am currently completing my Masters degree in Computer Science Education.

My student number is 891288849

Details of my supervisor are as follows:

Dr. D.W. Govender: Tel : 031 2603428

Email: govenderd50@ukzn.ac.za

In order to complete my degree, I need to conduct a research-study based on my chosen field of research at the school.

My research study explores the perceptions and experiences of teachers' and learners' when pair programming in the teaching and learning of programming. My research title is:

"A Study of using a cooperative learning technique when teaching object oriented programming".

I would like to request for permission to conduct my research at the above school.

I thank you in anticipation.

RESEARCHER: MR T.P Govender
CONTACT NUMBERS: 0845743807
70 Pastoral road

Asherville
Durban
4091

The Principal

Sir/Madam

I am a Masters of Education student at the University of Kwa-Zulu Natal. My research study is on “A Study of using a cooperative learning technique when teaching object oriented programming”.

The outcome of this study should provide valuable information which will contribute to the effectiveness of using pair programming as a teaching tool in an IT classroom. Observation of pedagogy and the information requested for, in the interview schedule and questionnaires, is significant to this study.

Thus, I seek the approval to visit your school, to observe teachers’ lessons for a day, hand out a questionnaire to learners and teachers, and thereafter at a later date, be allowed to interview the same educators whose lessons I have observed, during a time that is convenient to him/her. No individual or school will be identified in this study. Confidentiality is ensured.

I look forward to a favourable response.

Yours sincerely

RESEARCHER : T.P Govender
CONTACT DETAILS: 0845743807
031 208 8282

DATE: 15 November 2011

APPENDIX F: INTERVIEW SCHEDULE

15 November 2011

Introduction

The researcher starts off by introducing herself and explaining the purpose of the interview and the origins of her interest in the research topic, and to ask for permission to tape-record the interview as well as assuring confidentiality of information emanating from the interview.

- purpose of interview
- background to the study
- permission to tape-record
- reassurance about confidentiality of information

1. How many classes to you teach/lecture programming in your school?
2. How many students per class?
3. Have you received any training to teach programming?
4. What support have you received from teachers, management in your school, and from surrounding schools?
5. How did the “pair programming” improve your teaching?
6. What stage would you say you have reached in using the pair programming?
7. How did the concepts of pair programming improve your ability as a teacher to manage the classroom?
8. Does the use of pair programming speed up the pace of your lessons?
9. Does the use of pair programming aid you in assisting slow learners? How?

10. How does the use of pair programming affect the planning of your lessons and assessments with regard to time constraints?
11. What challenges did you experience initially when using pair programming in your classroom? How have you overcome them?
12. Are you currently experiencing any further challenges? List them.
13. Is there a difference with regard to learners' attentiveness in the lesson when you use "pair programming" to teach?
12. In your opinion, does a pair programming help learners to better learn programming concepts?
13. In your opinion when the learners are able to interact with their peers, does this affect their learning?
14. In your opinion how did the use of pair programming impact on learners' interest, enthusiasm and motivation, especially the lower achieving ones?
15. In what ways if any does pair programming address the following modalities of learning:
visual and auditory?
Visual: _____
Auditory: _____
16. What advice/recommendations would you give to a teacher who is a novice "pair programmer"?

APPENDIX G: LETTER TO PRINCIPAL

70 Pastoral Road,
Asherville, DURBAN 4091
Telephone: 031-2088282 / 0845743807
Fax: 0866741094

23 April 2012

The Principal

Dear Sir

RESEARCH: A Study of using a cooperative learning technique when teaching object oriented programming.

I am currently conducting a research project on the professional development of Information Technology (IT) teachers. SANPAD (South Africa-Netherlands Research Programme on Alternatives in Development) has awarded us a grant in order to do research in schools that have IT as one of their electives.

Professional development is an area that has for a long time been in need of adequate, practice orientated research. With the implementation of the new Curriculum and Assessment Policy Statement, that will replace the National Curriculum Statement teachers are in dire need of support in terms of content knowledge as well as pedagogical skills.

The KZN Department of Education has already granted permission to conduct the research in the province's schools. I have also informed the different district managers of this project.

I hereby kindly request you to allow your Gr. 11 IT-teacher to be involved in the research that will be conducted in April-May 2012. The teachers will first be interviewed in order to establish their needs. This will be followed by several professional development sessions for IT teachers which will not interrupt teaching in any way as we will schedule individual meetings with IT teachers when it suits

them. We will also ask permission from parents to allow Grade 11 learners to complete short questionnaires to determine their needs and skills regarding computer programming. All results will be available to you and your teacher upon request and will further be used to develop a model for professional development of IT teachers in South Africa. I am convinced that this research will contribute to more effective IT education in South Africa and will also strengthen the relationship between the university and schools.

I will contact your IT teacher shortly to arrange a first meeting.

Kind regards

Prinavin Govender

APPENDIX H: ETHICAL CLEARANCE LETTER DOE (KZN)



kzn education

Department:
Education
KWAZULU-NATAL

Enquiries: Sibusiso Alwar

Tel: 033 341 8610

Ref.:2/4/8/120

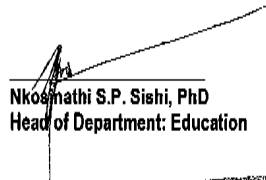
Mr. Thamotharan Prinavin Govender
70 Pastoral Road
Asherville
Durban
4091

Dear Mr. Govender

PERMISSION TO CONDUCT RESEARCH IN THE KZN DoE INSTITUTIONS

Your application to conduct research entitled: **A Study using a Cooperative Learning Technique when Teaching Object Oriented Programming (OOP)**, in the KwaZulu-Natal Department of Education Institutions has been approved. The conditions of the approval are as follows:

1. The researcher will make all the arrangements concerning the research and interviews.
2. The researcher must ensure that Educator and learning programmes are not interrupted.
3. Interviews are not conducted during the time of writing examinations in schools.
4. Learners, Educators, Schools and Institutions are not identifiable in any way from the results of the research.
5. A copy of this letter is submitted to District Managers, Principals and Heads of Institutions where the intended research and interviews are to be conducted.
6. The Period of investigation is limited to the period from 01 September 2011 to 01 September 2012.
7. Your research and interviews will be limited to the schools you have proposed and approved by the Head of Department. Please note that Principals, Educators, Departmental Officials and Learners are under no obligation to participate or assist you in your investigation.
8. Should you wish to extend the period of your survey at the school(s), please contact Mr. Alwar at the contact numbers below.
9. Upon completion of the research, a brief summary of the findings, recommendations or a full report / dissertation / thesis must be submitted to the research office of the Department. Please address it to The Director-Resources Planning, Private Bag X9137, Pietermaritzburg, 3200.
10. Please note that your research and interviews will be limited to the following Schools and Institutions:
 - 10.1 Hillgrove Secondary School
 - 10.2 Yukulakhe Secondary School
 - 10.3 Brindhaven Secondary School


Nkosinathi S.P. Sishi, PhD
Head of Department: Education

2011-12-04
Date

...dedicated to service and performance
beyond the call of duty.

KWAZULU-NATAL DEPARTMENT OF EDUCATION

POSTAL : Private Bag X9137, Pietermaritzburg, 3200, KwaZulu Natal, Republic of South Africa

PHYSICAL : Office G 25, 188 Pietermaritzburg Street, Metropolitan Building, Pietermaritzburg 3201

TEL. : Tel: +27 33 341 8610 | Fax: +27 33 3341 8612 | E-mail: sibusiso.alwar@kzndoe.gov.za |

Web: www.kzndoe.gov.za

No. 9914 P. 1

8. Dec. 2011 18:48

APPENDIX I: ETHICAL CLEARANCE - DUT



*Directorate for Research and Postgraduate Support
Durban University of Technology
Tromso Annexe, Steve Biko Campus
P.O. Box 1334, Durban 4000
Tel.: 031-3732576/7
Fax: 031-3732946
E-mail: moyos@dut.ac.za*

20th February 2012

Mr. T. P. Govender
c/o Department of Information Technology
Durban University of Technology

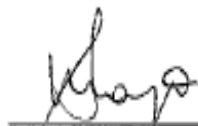
Dear Mr. T. P Govender

PERMISSION TO CONDUCT RESEARCH AT THE DUT

Your email correspondence in respect of the above refers. I am pleased to inform you that the Institutional Research Committee (IRC) will grant permission to you to conduct your research at the Durban University of Technology.

We would be grateful if a summary of your key research findings can be submitted to the IRC on completion of your studies.

Kindest regards,
Yours sincerely

A handwritten signature in black ink, appearing to read "S. Moyo", is written over a horizontal line.

PROF. S. MOYO
DIRECTOR (ACTING): RESEARCH AND POSTGRADUATE SUPPORT

APPENDIX J: ETHICAL CLEARANCE – UKZN



Research Office, Govan Mbeki Centre
Westville Campus
Private Bag x54001
DURBAN, 4000
Tel No: +27 31 260 8350
Fax No: +27 31 260 4609
snymanm@ukzn.ac.za

25 January 2012

Mr TP Govender (89128849)
School of Science, Maths & Technology

Dear Mr Govender

PROTOCOL REFERENCE NUMBER: HSS/0013/012M

PROJECT TITLE: A study of using a cooperative learning technique when teaching object oriented programming

In response to your application dated 22 November 2011, the Humanities & Social Sciences Research Ethics Committee has considered the abovementioned application and the protocol has been granted **FULL APPROVAL**.

Any alteration/s to the approved research protocol i.e. Questionnaire/Interview Schedule, Informed Consent Form, Title of the Project, Location of the Study, Research Approach and Methods must be reviewed and approved through the amendment /modification prior to its implementation. In case you have further queries, please quote the above reference number.

PLEASE NOTE: Research data should be securely stored in the school/department for a period of 5 years.

I take this opportunity of wishing you everything of the best with your study.

Yours faithfully

.....
Professor Steven Collings (Chair)
HUMANITIES & SOCIAL SCIENCES RESEARCH ETHICS COMMITTEE

cc. Supervisor – Dr DW Govender
cc. Mrs S Naicker / Mr N Memela