Microworlds, Poetry and Programming:

Block-Based Programming for Literacy and Computational Thinking in Secondary Education in Wales

Craig Jenkins

A submission presented in partial fulfilment of the requirements of the University of South Wales/Prifysgol De Cymru for the degree of Doctor of Philosophy

January 2018

i Abstract

In the 1960s, Papert and a team at the Massachusetts Institute of Technology (MIT) developed Turtle Graphics using the LOGO programming language. Underpinning this development was a profound new philosophy of how learning happens with computers using a **microworlds**-based approach to learning. The aim of Turtle Graphics was simple: through the act of learning to program an on-screen Turtle, learners also develop their conceptual understanding in other domains such as mathematics.

In contemporary Wales, literacy and numeracy figure as national priorities for school improvement. A statutory framework makes these skills a responsibility of all teachers, regardless of subject discipline or age range. The Welsh Government has stated that Digital Competence is set to join literacy and numeracy as a third cross-curricular responsibility with increasing prominence being given to thinking computationally.

This work builds upon the well-established theoretical position of microworlds research and contributes new knowledge to this area in two key ways. First, it considers the potential for reviving the microworld tradition in light of the aspirations and priorities within current education policy in Wales. Second, it considers how extensible block-based programming technologies may be used by school-based educators for developing their own activities containing custom block-kits.

Specifically, this project explores three key questions: (i) What is the potential of a microworld-based teaching approach for the development of specific aspects of computational thinking? (ii) What is its potential for defined aspects of literacy? (iii) What effective practice recommendations for microworld pedagogic design can be made?

The work comprises five studies, four of which took place at a school where the teacher-researcher is employed as a full time teacher of ICT. The project employs an action research approach that tells the story of five work packages (WP) that took place over six academic terms at the school. The WPs makes use of a range of methodological instruments including quantitative-based quasi-experiments and qualitative peer talk analyses. The project investigates the three key questions identified above by establishing a series of microworld-based interventions in the teaching schemes of eleven-to-twelve-year-olds.

The conclusions are limited by difficulties in applying rigid scientific methodologies in an educational environment, in particular the lack of random assignment of subjects and small sample sizes. However, quantitative findings suggest that learners undertaking microworld-based interventions made greater improvements than their comparison group counterparts in the specific aspects of literacy and computational thinking that were tested for. There is also some evidence to indicate a link in performance between literacy and computational thinking, but this link requires more robust evidence. Further, nine effective practice recommendations are put forward for educators wishing to replicate microworld-based pedagogy in their own practice such as the merits of pair programming and dialogic teaching.

ii Contents

I ABSTRACT 2

II CONTENTS 4

III LIST OF TABLES 8

IV LIST OF FIGURES 9

V ABBREVIATIONS 10

VI ACKNOWLEDGEMENTS 10

1 INTRODUCTION 11

1.1 BACKGROUND 11
1.1.1 THE PEDAGOGY OF THE MICROWORLD 11
1.1.2 EARLY MICROWORLDS FOR LITERACY 12
1.1.3 BLOCK-BASED PROGRAMMING LANGUAGES 13
1.1.4 FOCUSING ON LITERACY AND COMPUTATIONAL THINKING IN WALES 13
1.2 RATIONALE AND STATEMENT ON ORIGINALITY 14
1.3 ACADEMIC EXCHANGE 16
1.4 OVERVIEW 16
1.4.1 CHAPTER 2 – LITERATURE REVIEW 16
1.4.2 CHAPTER 3 - METHODOLOGY 18
1.4.3 CHAPTER 4, WORK PACKAGE 1: BYOB CUSTOM BLOCK KITS ACROSS NATIONAL CURRICULUM AREAS 18
1.4.4 CHAPTER 5, WORK PACKAGE 2: USING THE POEM GENERATOR CUSTOM BLOCK KIT IN ENGLISH LESSONS 19
1.4.5 CHAPTER 6, WORK PACKAGE 3: POEMS WITH THE SNAP! AND SMALL BASIC LANGUAGES 19
1.4.6 CHAPTER 7, WORK PACKAGE 4: PAIRED PROGRAMMING WITH SNAP! 19
1.4.7 CHAPTER 8, WORK PACKAGE 5: BRINGING LITERACY TO AN EXTRACURRICULAR CLUB 20
1.5 GUIDING QUESTIONS 20
1.6 OVERVIEW OF FINDINGS 21

2 REVIEW OF THE LITERATURE 23

2.1 THE DEVELOPMENT OF CONSTRUCTIONISM IN PROGRAMMING EDUCATION 23
2.1.1 THE COMPUTER AS TUTOR, TOOL, TUTEE 23
2.1.2 INSTRUCTIONISM: THE COMPUTER AS TUTOR 25
2.1.3 PIAGET AND CONSTRUCTIVISM 26
2.1.4 PAPERT AND CONSTRUCTIONISM 29
2.1.5 THEMES IN CONSTRUCTIONISM I: LEARNING BY DESIGN 30
2.1.6 THEMES IN CONSTRUCTIONISM II: MULTIPLE WAYS OF KNOWING 31
2.1.7 THEMES IN CONSTRUCTIONISM III: REIMAGINING LEARNING COMMUNITIES 32
2.1.8 PIAGET VERSUS VYGOTSKY: THE COMPETING ROLES OF THE INDIVIDUAL AND THE SOCIAL 34
2.1.9 TEACHING AND LEARNING WITH MICROWORLDS 36
2.1 PAPERT, LOGO AND THE ADVENT OF TURTLE GRAPHICS 39
2.2.1 INTRODUCING THE TURTLE GRAPHICS MICROWORLD 39
2.2.2 THE SIGNIFICANCE OF PLAYING TURTLE 41
2.2.3 THE MATHETICS OF TURTLE GRAPHICS 43
2.2.4 PROBLEMATIZING PAPERT 46
2.3 MOVING FURTHER WITH MICROWORLDS: RESNICK, STARLOGO AND RIEBER 48
2.3.1 DECENTRALIZED SYSTEMS AND SELF-ORGANISING BEHAVIOURS 48
2.3.2 STARLOGO: A MICROWORLD FOR EXPLORING COMPLEXITY-FROM-SIMPLICITY SITUATIONS 50

2.3.3 RIEBER'S FRAMEWORK FOR MICROWORLDS 52
2.4 CODING AS A CREATIVE ENTERPRISE 55
2.4.1 BRICKLAYING TO PLAYING THE CELLO: AN INTIMATE CONNECTION BETWEEN HAND AND HEAD 55
2.4.2 MAKING IS CONNECTING 58
2.4.3 CREATIVE PROGRAMMING PROJECTS 62
2.5 BLOCK-BASED PROGRAMMING APPROACHES 65
2.5.1 DEFINING PROGRAMMING 66
2.5.2 PROGRAMMING WITH SCRATCH 67
2.5.3 ROBOTICS AND PLAYFUL LEARNING 74
2.5.4 ENGAGING GIRLS WITH PROGRAMMING 75
2.5.5 'LOWERING THE FLOOR' OF PROGRAMMING 77
2.6 DEVELOPING COMPUTATIONAL THINKING AS A STRAND OF DIGITAL COMPETENCE 81
2.6.1 SURVEYING THE POLITICAL LANDSCAPE OF ICT EDUCATION IN WALES 81
2.6.2 LEARNING TO THINK THROUGH LEARNING TO PROGRAM 84
2.6.3 ALGORITHMIC THINKING 86
2.6.4 A MODEL OF COMPUTATIONAL THINKING 86
2.7 EXPLORING THE RELATIONSHIP BETWEEN LITERACY AND COMPUTATIONAL THINKING 88
2.7.1 SURVEYING THE POLITICAL LANDSCAPE OF LITERACY IN WALES 88
2.7.2 LITERACY AND LANGUAGE IN USE 90
2.7.3 MICROWORLDS FOR LANGUAGE 92
2.7.4 MICROWORLDS FOR POETRY 94
2.7.5 POETS AND PROGRAMMERS 96

3 METHODOLOGICAL OVERVIEW 98

3.1 THE ACTION RESEARCH APPROACH 98
3.2 A MIXED-METHODS APPROACH 100
3.2 INSTRUMENTS 101
3.2.1 QUANTITATIVE INSTRUMENTS 101
3.2.2 OBSERVATION 103
3.2.3 SURVEY DATA 104
3.3 MAXIMIZING CREDIBILITY 104
3.3.1 DATA TRIANGULATION 105
3.3.2 INVESTIGATOR TRIANGULATION 105
3.3.3 THEORETICAL TRIANGULATION 106
3.3.4 METHODOLOGICAL TRIANGULATION 106
3.4 DATA ANALYSES 107
3.4.1 QUALITATIVE ANALYSES 107
3.4.2 QUANTITATIVE ANALYSES 108
3.5 SAMPLE 109
3.5.1 SCHOOL 1 IN CONTEXT (WP1, 3, 4, 5) 109
3.5.2 SCHOOL 2 IN CONTEXT (WP2) 111
3.5.3 SAMPLING CONSIDERATIONS 112
3.6 ETHICAL CONSIDERATIONS 113
3.6.1 SCHOOL LEADERS 113
3.6.2 TEACHERS 114
3.6.3 PARENTS/CARERS 114
3.6.4 LEARNERS 114

4 WORK PACKAGE 1: BYOB CUSTOM BLOCK KITS ACROSS NATIONAL CURRICULUM AREAS 116

4.1 INTRODUCTION 116
4.1.1 OVERVIEW OF THE MOODLE COURSE ACTIVITIES 117
4.1.2 FORMAT OF THE MOODLE QUIZ TOOL 118
4.1.3 DESIGN OF THE MICROWORLD ACTIVITIES 118

4.2 Methods **125**
4.2.1 Format of the Moodle Database Learning Journals 126
4.3 Results and Discussion **127**
4.4 Conclusions and Limitations **130**

5 WORK PACKAGE 2: THE POEM GENERATOR CUSTOM BLOCK KIT IN ENGLISH LESSONS **132**

5.1 Introduction **132**
5.1.1 Design of the Custom Snap! Block-Kit 132
5.2 Methods **135**
5.2.1 Design of the Pre-tests and Post-tests 137
5.2.2 Participants 138
5.3 Results and Discussion **139**
5.3.1 Comparison of Means 140
5.3.2 Significance Testing 140
5.3.3 Forming a Regression Model 141
5.4 Conclusions and Limitations **142**

6 WORK PACKAGE 3: MAKING POEMS WITH THE SNAP! AND SMALL BASIC LANGUAGES **144**

6.1 Introduction **144**
6.1.1 Design of the Small Basic Microworld 144
6.2 Methods **146**
6.2.1 Design of the Teaching Scheme 147
6.2.2 Design of the Pre-tests and Post-tests for the Quantitative Analysis 148
6.2.3 Design of the Survey for the Qualitative Analysis 149
6.3 Results and Discussion **151**
6.3.1 Quantitative Results 151
6.3.2 Qualitative Survey Data 155
6.4 Conclusions and Limitations **159**

7 WORK PACKAGE 4: PAIRED PROGRAMMING WITH SNAP! **162**

7.1 Introduction **162**
7.2 Methods **162**
7.2.1 Design of the Teaching Scheme 162
7.2.2 Peer Talk Analysis 164
7.2.3 Pre-tests and Post-tests 165
7.3 Results and Discussion **165**
7.3.1 Quantitative Results 165
7.3.2 Qualitative Results 168
7.4 Conclusions and Limitations **171**

8 WORK PACKAGE 5: BRINGING LITERACY TO AN EXTRACURRICULAR CLUB **174**

8.1 Introduction **174**
8.1.1 Phase 1 – Interactive Tutorials 175
8.1.2 Phase 2 – Tanaga Challenge 176
8.2 Methods **177**
8.3 Results and Discussion **179**
8.3.1 Overview 179
8.3.2 Analysis of Individual CE 181
8.4 Conclusions and Limitations **190**

9 CONCLUSIONS **194**

9.1 Limitations **194**

9.1.1 SAMPLING LIMITATIONS 194
9.1.2 METHODOLOGICAL LIMITATIONS 195
9.2 KEY FINDINGS 196
9.2.1 GQ1. COMPUTATIONAL THINKING 196
9.2.2 GQ2. ASPECTS OF LITERACY 198
9.2.3 GQ3. RECOMMENDATIONS FOR THE CLASSROOM 200
9.2.4 OTHER FINDINGS 205
9.3 IMPLICATIONS 206
9.3.1 CHANGING CLASSROOM PRACTICE 206
9.3.2 POLICY DEVELOPMENTS IN WALES 207
9.4 FUTURE WORK 209

REFERENCES 210

APPENDICES 225

APPENDIX A: MICROWORLDS ACTIVITY PACK AND BYOB DATA FILES FOR WP1 226
APPENDIX B: SMALL BASIC MICROWORLD PROGRAMMING CODE AND DATA FILE FOR WP3 AND WP4 227
APPENDIX C: ETHICAL CONSENT FORMS 228
APPENDIX C1 – LEARNER AND PARENT/CARER CONSENT WP2, WP3, WP4 229
APPENDIX C2 – TEACHER CONSENT WP2, WP3, WP4 230
APPENDIX C3 – LEARNER AND PARENT/CARER CONSENT WP5 232
APPENDIX D: WORK PACKAGE 2 INTERVENTION 233
APPENDIX D1 –WP2 TEACHER INSTRUCTION SHEET 234
APPENDIX D2 – ABRIDGED WP2 COMPUTATIONAL THINKING TEST QUESTIONS 236
APPENDIX D3 – ABRIDGED WP2 LITERACY TEST QUESTIONS 237
APPENDIX E: WORK PACKAGE 3 INTERVENTION 238
APPENDIX E1 – ABRIDGED WP3 COMPUTATIONAL THINKING TEST QUESTIONS 239
APPENDIX E2 – ABRIDGED WP3 LITERACY TEST QUESTIONS 240
APPENDIX F: WORK PACKAGE 4 INTERVENTION 241
APPENDIX F1 – ABRIDGED WP4 'UNPLUGGED' COMPUTATIONAL THINKING ACTIVITIES 242
APPENDIX F2 – ABRIDGED WP4 'UNPLUGGED' LITERACY ACTIVITIES 243
APPENDIX G: WORK PACKAGE 5 EXTRACURRICULAR CLUB 244
APPENDIX G1 – SEDA SCHEME CA FREQUENCIES 245
APPENDIX G2 – ABRIDGED WP5 ACTIVITY BRIEF 246
APPENDIX G3 – ABRIDGED WP5 SUPPORTING RESOURCES 247
APPENDIX H: PROJECT OUTLINE DIAGRAM 248
APPENDIX H1 – PROJECT OUTLINE DIAGRAM 249

iii List of Tables

TABLE 1: COMPUTATIONAL OBJECTS IN TURTLE GRAPHICS AND STARLOGO 51
TABLE 2: COMPUTATIONAL OBJECTS IN TURTLE GRAPHICS, STARLOGO & SCRATCH 70
TABLE 3: DRAFT DIGITAL COMPETENCE FRAMEWORK EXCERPT OF EXPECTATION STATEMENTS 83
TABLE 4: LITERACY FRAMEWORK EXCERPT OF EXPECTATION STATEMENTS 89
TABLE 5: KNOWLEDGE MAPPING OF ROCKET MAZE 119
TABLE 6: KNOWLEDGE MAPPING OF TRAFFIC LIGHT SIMULATOR 120
TABLE 7: KNOWLEDGE MAPPING OF TWINKLE LITTLE STAR 121
TABLE 8: KNOWLEDGE MAPPING OF FEUDAL SOCIETY 122
TABLE 9: KNOWLEDGE MAPPING OF PYRAMID OF NUMBERS 122
TABLE 10: KNOWLEDGE MAPPING OF SENTENCE GENERATOR 124
TABLE 11: KNOWLEDGE MAPPING OF MUSICAL PICTURES 125
TABLE 12: PERCENTAGE OF LEARNERS IDENTIFYING AN IMPROVEMENT IN SUBJECT-SPECIFIC KNOWLEDGE 128
TABLE 13: DESCRIPTION OF THE CUSTOM BLOCK KIT IN SNAP! 133
TABLE 14: ASSESSMENT MATRIX OF LITERACY EXPECTATION STATEMENTS 138
TABLE 15: ASSESSMENT MATRIX OF COMPUTATIONAL THINKING CONCEPTS AND PRACTICES 138
TABLE 16: WP2 INTERVENTION COMPARISON OF MEANS 140
TABLE 17: SUBROUTINES IN THE SMALL BASIC MICROWORLD 145
TABLE 18: LITERACY ASSESSMENT MATRIX 148
TABLE 19: COMPUTATIONAL THINKING ASSESSMENT MATRIX 149
TABLE 20: WP3 INTERVENTION COMPARISON OF MEANS 152
TABLE 21: WP4 COMPARISON OF MEANS 166
TABLE 22: BLOCKS AND THEIR ACTIONS 177
TABLE 23: SEDA CLUSTER AND CODE SUMMARY 179
TABLE 24: DIALOGUE CE1 181
TABLE 25: DIALOGUE CE2 182
TABLE 26: DIALOGUE CE3 183
TABLE 27: DIALOGUE CE4 183
TABLE 28: DIALOGUE CE5 184
TABLE 29: DIALOGUE CE6 185
TABLE 30: DIALOGUE CE7 186
TABLE 31: DIALOGUE CE8 187
TABLE 32: DIALOGUE CE9 188
TABLE 33: DIALOGUE CE10 188
TABLE 34: DIALOGUE CE11 190
TABLE 35: DIALOGUE CE12 190

iv List of Figures

FIGURE 1: POSITIONING A PAPERTIAN LEARNING APPROACH ____ 35
FIGURE 2: THE SCRATCH USER INTERFACE ____ 68
FIGURE 3: ARRANGEMENT OF PROGRAMMING BLOCKS IN SCRATCH ____ 69
FIGURE 4: PICOBOARD USED IN WP1 (THE PLAYFUL INVENTION COMPANY, 2010B) ____ 74
FIGURE 5: CREATING A CUSTOM BLOCK ____ 78
FIGURE 6: SCREEN GRAB OF ONE TOPIC IN THE MOODLE VLE COURSE ____ 116
FIGURE 7: A MATCHING QUESTION FOR THE CURRICULUM AREA OF BIOLOGY ____ 118
FIGURE 8: SCREEN GRAB OF SENTENCE GENERATOR ____ 123
FIGURE 9: LEARNING JOURNALS USING A MOODLE DATABASE ____ 126
FIGURE 10: PERCENTAGE OF LEARNERS IDENTIFYING IMPROVEMENTS IN COMPUTATIONAL THINKING ____ 127
FIGURE 11: THE SCRIPTS AREA OF THE POEM GENERATOR IN SNAP! ____ 134
FIGURE 12: THE STAGE AREA OF POEM GENERATOR IN SNAP! ____ 135
FIGURE 13: A SURVEY USING SURVEYMONKEY.COM ____ 150
FIGURE 14: COMPUTATIONAL THINKING DEVELOPMENT FREQUENCY CHART ____ 154
FIGURE 15: LITERACY DEVELOPMENT FREQUENCY CHART ____ 154
FIGURE 16: COMPARISON OF CA FREQUENCY FOR LEARNERS AND TEACHERS BY CODE CLUSTER ____ 180
FIGURE 17: BREAKDOWN OF CA CODES IN CLUSTER R – MAKE REASONING EXPLICIT FOR LEARNERS ____ 180
FIGURE 20: ETHAN’S TANAGA POEMS ____ 184
FIGURE 21: ETHAN’S CUSTOM BLOCK ____ 185
FIGURE 22: ROKAS’ LINES OF POETRY ____ 189
FIGURE 23: NAVIGATING THE DATA DISC ____ 225
FIGURE 24: DATA DISC DIRECTORY ____ 225

v Abbreviations

Abbreviation	Meaning
GQ	Guiding question
WP	Work package
KS3	Key Stage 3 (The third stage of compulsory school age education in Wales that is comprised of learners aged between eleven and fourteen. KS3 is in turn comprised of three 'year' groups known as year seven, year eight and year nine).
eFSM	Free school meal eligibility (A measure of socioeconomic deprivation).
ICT	Information and Communications Technology
R	Recommendation
UI	User interface
AoLE	Area of Learning and Experience
DCF	Digital Competence Framework
LNF	Literacy and Numeracy Framework
CPD	Continuing professional development
NC	National Curriculum
STEM	Science, technology, engineering and mathematics
SoL/SoW	Scheme of learning or scheme of work
BCS	British Computer Society
CAS	Computing at School Group
ALN	Additional Learning Need
SEN	Statement of Educational Need
ITE	Initial Teacher Education

vi Acknowledgements

Thanks to Torbjorn Dahl (University of Plymouth), Ceri Pugh (University of South Wales) and David Longman (independent academic) for their many helpful comments and insights. Further thanks to Gavin Jones, Victoria Corken, Dean Seabrook and Jemma Paull for giving up their time to support this research in what is always a very busy school day.

1 Introduction

In the 1960s, Papert and his team at the Massachusetts Institute of Technology (MIT) developed a programming language called LOGO. Underpinning this development was a profound new philosophy of how learning happens with computers: a **microworld-based** approach to learning. This action research project examines how secondary school teachers may revive a microworld-based approach in a contemporary educational context using extensible block-based programming technologies enabling custom block creation. It examines the microworld as a pedagogical tool in secondary level learning in Wales at Key Stage 3 (KS3) and explores its potential role in making defined aspects of literacy and computational thinking development accessible through practical approaches influenced by constructionism.

This introductory chapter begins with a summary of some of the key issues and debates surrounding the research area in order to identify a rationale for the thesis. Whilst this is intended as a useful summary, a more comprehensive account is provided in the literature review in chapter two. The background and rationale is followed by an outline of the thesis as a whole to guide the reader through its different chapters. Finally, this chapter ends by stating the guiding questions that connect each of the chapters together along with a summary of the key findings.

1.1 Background

The traditional *instructionist* model of learning with technology positions the computer as the active agent in knowledge construction where the learner is a recipient of knowledge (Al Khatib, 2009). Such a model might be characterised by so-called ‘drill-and-practice’ software packages (Papert, 1993a). In contrast, *constructivist* models of learning emphasize the active role of the learner in constructing their own mental models through exploratory learning experiences (Dalgarno, 2001). *Constructionism* develops the constructivist position further by adding that the making of some kind of meaningful *product*, such as a poem which is the focus of this investigation, is a particularly effective way of making *knowledge* (Resnick and Kafai, 1996).

1.1.1 The Pedagogy of the Microworld

The microworld-based learning approach provides a digital space to make guided discovery possible (Rieber, 1992). A microworld enables the learner to make things – and learn things - through careful design of the digital space. The educator decides the boundaries of the microworld and therefore constrains the learning experience to enable exploration within a particular domain of knowledge. A microworld bridges the gap between instructionism and constructionism by recognising the value of exploratory learning, but only within boundaries that are defined by the educator.

Papert (1980a) developed the quintessential example of a microworld: LOGO Turtle Graphics. Turtle Graphics is an environment to enable learners to write their own computer programs for drawing on-screen graphics. Learners type four key instructions; to move forward, back, left and right; as they control an on-screen turtle with a pen attached to it. One of the key principles of Turtle Graphics is that, through the act of programming the computer, learners make mathematical discoveries such as geometric angle of turn (Papert, 1980c).

Papert made bold claims about his microworld, not least that learners can improve their problem solving skills – in addition to their geometric/mathematical skills - by programming within it (1980a). Studies such as that by Simmons et al. (1993) have questioned such claims by instead pointing to gains resulting from careful teacher-mediated interventions. It seems, then, that careful consideration needs to be given to the pedagogy of a microworld and not merely its use as a technological tool, which is considered more fully in the literature review.

1.1.2 Early Microworlds for Literacy

In the 1980s, Papert's Turtle Graphics microworld provided an environment for guided discovery within the domain of mathematics. Sharples (1985) created a series of exploratory activities as part of a teaching scheme for written English. In order to establish the efficacy of the teaching scheme, Sharples carried out teacher assessments and feature analyses of essays produced by learners before and after the intervention. Though the gains were not statistically significant, the experimental group post-essays revealed a greater increase in the development of 'mature' writing features than the control group (*ibid.*, pp. 103 - 104). This provides modest evidence that thinking computationally – through learning to program – could potentially have a positive impact upon literacy development.

Sharples' first program was PAT: a word pattern generator. Learners develop whole sentence structures, that can be repeated, with a combination of textual programming and picks from random word lists. Ideas similar to this one by Sharples are also developed in Goldenberg and Feurzeig's (1987) experiments with LOGO for language. Particularly interesting here was the experiment focusing on how programming tools may be applied within the domain of poetry. Goldenberg and Feurzeig's experiments, though useful, were not subjected to quantitative instruments in an educational setting.

1.1.3 Block-based Programming Languages

A potential problem with some early microworlds was that they were *text-based* which added the barrier of syntax (Resnick, 2009) to programming a computer. This can be problematic for a microworld since Rieber (2004) suggests that a 'low floor' is required to increase its accessibility to a large group of learners. Scratch (Resnick et al., 2009) popularized a new mode of learning to program: by snapping together programming blocks in a way can be likened to as assembling virtual LEGO bricks. An important point here is that the shape of the blocks themselves help to prevent syntactic errors from occurring. This issue that once faced novice programmers when working with text was removed and the accessibility of programming widened.

What makes the block-based approach interesting for educators is the possibility to design a set of accessible microworld activities, using custom programming block-kits, effectively bespoke programming languages, according to the domain of knowledge to be developed in the lesson. In music lessons, a custom block kit can be created to produce a melody using blocks such as 'play note for seconds'. In history, a custom block kit can be used to model the feudal system and experiment with changing numbers of kings, barons, knights and peasants. In English, a poem generator can be used to construct poems with custom 'write line' and 'pick from word list' blocks. These very ideas are developed in WP1.

Scratch 2.0 (MIT Media Lab, 2016) enables such extensibility of custom block creation, though it is the Snap!/BYOB (Build Your Own Blocks) (Mönig and Harvey, 2009) extension that is the original platform for custom-block building with added functionalities. Snap! enables user-created programming blocks to be placed within other user-created programming blocks, for example.

1.1.4 Focusing on Literacy and Computational Thinking in Wales

Sharples' (1985) text-based programming activities in LOGO are aligned with the *language in use* approach to teaching written English that has remained the dominant paradigm of language teaching to the present day. There has been move from so-called 'drill and practice' grammar exercises (Ellis, 2003) towards a model of language teaching that provides learners with specific tasks for meaningful written communication (*see* Richards and Reppen, 2014; Fleming, 2006). Literacy is concerned with the application of these language skills to meaningful communication with a useful *context* (Cambridge Assessment, 2013).

Educational policy in Wales has focused on improving literacy within its schools for some time. Following concerns by employers (Future Skills Wales, 2004) and a review of the curriculum in Wales (ACCAC, 2004), the Welsh Government (2008i) announced that *communication* would become a non-statutory skill for development across the curriculum in Wales. The emphasis on literacy has intensified even further in recent years. The inspectorate for schools in Wales, Estyn (2012) said that an even greater focus on literacy was required. The Welsh Government (2011) responded by designating literacy a statutory requirement for all teachers across subject areas through the introduction of the Literacy and Numeracy Framework (LNF).

Running alongside the increased prominence that is being afforded to literacy in Wales is the changing nature of ICT as a subject. In England, ICT has been removed in favour of a more programming-rich Computer Science (DfE, 2013) programme of study. In Wales, there have been calls to increase programming activities in the curriculum from the ICT Steering Group (2013) and a curriculum review (Welsh Government, 2015a).

More recently, the Welsh Government (2016b) has released a draft Digital Competence Framework (DCF) that designates *digital* skills across the curriculum an additional cross-curricular responsibility for all school age teachers. A strand of this framework has been dedicated to data and computational thinking with elements of programming education.

There is an increased interest in programming activities as well as its broader relevance across the curriculum, for example through the increased emphasis given to the idea of 'computational thinking' (Wing, 2006; DfE, 2013) and how this figures as part of wider digital competence (Welsh Government, 2016b). This should lead us to reconsider the pedagogy of both programming and its use as a tool for thought. It is therefore timely to investigate again the pedagogy of microworlds and revive such research in the context of the contemporary curriculum and programming technologies.

1.2 Rationale and Statement on Originality

Ian Willis (2013), writing for a doctoral research website at the University of Liverpool, provides a useful definition of the concept of *originality* that should be integral to Ph.D study: '[originality] opens up neglected areas or takes a new viewpoint on an old problem'. This project uses well-established methods in educational research and approaches to theoretical areas that have attracted a great deal of attention. It extends this background through a reimagining of the microworld-based approach by (i) situating it within a new curriculum context of contemporary Wales and (ii) using new extensible block-based programming tools that were not available during the microworlds research of the 1980s.

The originality of this project, however, moves beyond the revival of a previous theoretical position. It is essential that up-to-date, classroom-based research is conducted that examines programming activities with two particular foci on both cross-curricular literacy and computational thinking development.

Given its increased prominence as a national priority for school improvement and its statutory designation as a responsibility of all teachers, literacy figures as an important area of cross-curricular development (DfES, 2011; Welsh Government, 2011). Computational thinking, meanwhile, is now higher on the educational agenda in Wales than at any time since the early 1980s with the introduction of the draft Digital Competence Framework (Welsh Government, 2016b). It is surprising, then, that there is a clear absence of research coverage regarding how a school might integrate literacy and computational thinking development. A microworld-based approach provides one possible way of addressing this need for cross-curricular integration.

A great deal of the empirical work carried out here draws on Sharples (1985) in terms of the pedagogic design of the microworld and the focus on language. What it adds to Sharples' work, however, is it that is deconstructs the gains made in computational thinking into specific computational practices. The project extends our knowledge by providing new evidence relating to how block-based language microworlds may build computational thinking *concepts* such as abstraction (*see* section 8.3.2.iii), iteration (*see* ibid.) and data containers (*see* ibid.) alongside computational thinking *practices* such as testing and debugging (*see* section 8.3.2.iv) and reusing and remixing (*see* section 8.3.2.v). Further, a key methodological feature making this project distinct is the application of the fine-grained SEDA scheme for analysing classroom dialogue when working with microworlds (*see* WP5). The dual focus on developing specific aspects of literacy alongside deconstructed aspects of computational thinking is an important idea that addresses a key curriculum reform priority in Wales, namely for a greater focus on models of teaching that reinforce cross-curricular responsibilities (Donaldson, 2015).

The research paradigm employed in this project is an action research model that is particularly effective at bringing about positive change at a local, institutional level (*see,* Cohen, Manion and Morrison, 2011; Denscombe, 2002). The majority of the research presented here is situated within a single school where the teacher-researcher is employed as an ICT teacher. The most recent inspection report includes a recommendation that this school focuses on 'development of pupil's literacy and numeracy skills' (Estyn, 2016). Further, the report states that 'pupils do not use and develop their ICT skills well enough across the curriculum at key stage 3' (*ibid.*). It is clear, then, how the aims of the project align with school-level improvement priorities in addition to those identified at a national level.

This project aims to extend our knowledge of the pedagogy of computers in the curriculum beyond previous theoretical positions while building on the strengths and aspirations of current educational policy with respect to new approaches to computing and computer science. Whilst the study focuses on the secondary education in Wales, the findings reported are also applicable to educators beyond this.

1.3 Academic Exchange

The findings reported here have been subjected to scholarly peer review processes through published conference proceedings as well as an article in a published journal.

The initial concept of using extensible block-based languages within cross-curricular contexts was discussed at ICICTE (International Conference on Information Communication Technologies in Education) in 2012 (Jenkins, 2012). Initial self-reporting data based on these ideas were presented at the ITTE (Information Technology in Teacher Educator) conference in 2013 (Jenkins and Longman, 2013).

Quantitative findings derived from pre-tests and post-tests were delivered to ITTE in 2015 (Jenkins, 2015a) and appear in the published proceedings of the 10th WiPSCE (Workshop in Primary and Secondary Computer Education) conference (Jenkins, 2015b).

The findings from WP3 appear in IJEDICT (International Journal of Education and Development Using Information and Communication Technology (Jenkins, 2015c).

1.4 Overview

The following overview is designed to provide the reader with a structure for approaching the thesis (for a project outline, please also see appendix H). It begins with an overview of the literature review and methodology before moving to a series of summaries detailing the empirical work that comprises the five work packages (WP).

1.4.1 Chapter 2 – Literature Review

The literature review is divided into seven sections in order to access the broad and multidisciplinary literature that is considered. These sections are organized around seven themes; as discussed below.

Section 1. The Development of Constructionism in Programming Education

The literature review begins by considering key debates in school-age programming education. In particular, it examines the critiques of instructionism and looks in detail at the movement towards constructionist approaches that began in the 1980s.

Section 2. Seymour Papert, LOGO and the Advent of Turtle Graphics

This section examines Turtle Graphics as the quintessential example of the microworlds-based approach. It moves to examine the notion of the *turtle* more closely before considering some of the criticisms about Turtle Graphics.

Section 3. Moving Further with Microworlds: Resnick, StarLogo and Rieber

Moving on from Turtle Graphics, Resnick developed StarLogo as a microworld for exploring decentralized systems. After considering this development, the section moves to look at work by Rieber in order to provide a conceptual framework for understanding the microworlds-based learning approach more clearly.

Section 4. Coding as a Creative Enterprise

The act of programming a computer has traditionally been associated with STEM subjects, but this section begins by exploring the notion of *design*. Following this, research is considered that repositions programming beyond the STEM arena as a creative and generative activity.

Section 5. Introducing the Block-Based Programming Approach

In order to increase the accessibility of learning programming, a new building-block paradigm has entered programming education. This section examines Scratch as well as BYOB/Snap! that enables the extensibility of custom programming blocks.

Section 6. Developing Computational Thinking as a Strand of Digital Competence

This section begins with an examination of the contemporary policy context in programming education within both Wales and the UK. It then moves to explain the meaning of *computational thinking* and how this is developed within the overarching framework of *digital competence* in secondary education in Wales.

Section 7. Exploring the Relationship Between Literacy and Programming

The concluding section of the literature review examines previous studies where the microworlds-based pedagogic approach has been applied to the subject domain of written English. It then ends by exploring the relationship between programming and literacy more closely.

1.4.2 Chapter 3 - Methodology

The research presented here subscribes to an action research model (Cohen, Manion and Morrison, 2011) in order to bring about positive changes in programming education within the secondary school where the researcher is employed as a teacher of ICT.

The project uses a structure of five work packages (WP) that were guided by a cyclical approach: a 'spiral of planning' (Lewin, 1946). The WPs chronicle a development from the initial pilot of several microworld-based activities to a literacy-focused curriculum activity and teaching scheme over a number of terms. All five WPs (with the exception of the WP2) take place within a single secondary school setting. The research gains credibility as it is situated within a realistic classroom-based setting and seeks to address an authentic curriculum issue that exists within the organisation.

Across the five WPs, a mixed methods approach was used with a range of both quantitative and qualitative data collected.

The quantitative elements made use of quasi-experiments with comparison groups. Here, pre-tests and post-tests measured gain in aspects of literacy and computational thinking following the interventions characterized by a bespoke microworlds-based teaching scheme. Samples here were relatively small and, as these studies were based in a school and subject to timetabling constraints, not truly randomized. Quantitative data were subjected to significance testing and effect size calculations carried out using SPSS. The data collected using these instruments largely informed guiding question (GQ) 1 and GQ2 (*see* section 1.5 for a description).

Qualitative data were collected primarily from observation. Here, voice recordings were analysed using pre-existing coding frames (Mercer and Wegerif, 1997; Hennessy et al. 2016a) for peer talk in classroom settings. In WP3, learners completed an electronic learning journal following the microworld intervention. For this, open coding was used with a coding frame established to structure the findings. Qualitative data analyses contributed primarily to GQ3 (*see* section 1.5).

1.4.3 Chapter 4, Work Package 1: BYOB custom block kits across national curriculum areas

A wide range of activities constructed using the BYOB programming language were designed with links to a wide range of subject domains within the National Curriculum for Wales. The aim of this pilot study was to explore the potential of BYOB for building domain-specific microworld activities using the extensibility offered by custom block sets.

Activities were mapped to National Curriculum (NC) requirements in areas ranging from art and design and music to biology and mathematics. These activities were piloted within discrete ICT lessons and were led by the teacher-researcher of this project. Learners were asked to maintain a learning journal in order to self-report on their knowledge acquisition in terms of computational thinking and subject-specific knowledge.

1.4.4 Chapter 5, Work Package 2: Using the poem generator custom block kit in English lessons

A single microworld-based activity, created using the web-based iteration of BYOB known as Snap!, was used with an intervention and comparison group. The intervention focused much more narrowly on written English and computational thinking development.

This was the only study that took place outside the organisation where the researcher is employed as a teacher. A subject-specialist English teacher at School 2 led the intervention. A quasi-experimental design with non-equivalent groups was used in order to collect pre-test and post-test data for computational thinking and English subject-specific knowledge.

1.4.5 Chapter 6, Work Package 3: Making poems with the block-based Snap! and text-based Small Basic languages

A concurrent transformative (*see* Terrell, 2012) design was used for WP3 and WP4 of the project. This methodological orientation describes the way in which quantitative and qualitative data were collected concurrently. The research was guided, meanwhile, by the principles of the microworld-based theoretical perspective considered in the literature review.

The first *block-based* activity created using Snap! was supplemented with an additional *textual* programming activity created using Small Basic. The intervention focused on cross-curricular development in specific aspects of literacy in addition to computational thinking. A subject-specialist English and Drama teacher led the intervention. A non-equivalent group design was once again used to facilitate quantitative data collection. This was supplemented with a qualitative content analysis of survey/learning data from the intervention group.

1.4.6 Chapter 7, Work Package 4: Paired programming with Snap! and developing a microworlds-based teaching scheme for literacy and programming

This study duplicated WP3 in terms of the quantitative instruments with an additional group of learners. This was necessary in order to ensure that the positive results achieved from the previous study could be replicated. Greater attention was also paid here to the *pedagogic* design of the microworld rather than the *technical* design of the microworld itself.

This study adds to WP3 by making use of a new instrument of peer talk analysis. Voice recorded paired programming interactions were analysed using a limited coding scheme (Mercer and Wegerif, 1997) for classroom dialogue.

1.4.7 Chapter 8, Work Package 5: Bringing literacy to an extracurricular programming poetry club

Due to timetabling constraints at the school, it was felt necessary to make use of an extracurricular club format for WP5.

The freedom offered by this approach, without curriculum constraints, allowed for a larger volume of voice recording data to be collected. The resulting analysis uses a more comprehensive coding scheme (Hennessy et al., 2016a) and adds a more substantive qualitative dimension to evaluating the impact of the microworld-based activities.

1.5 Guiding Questions

In light of the rationale and overview provided above, three guiding questions (GQ) were formed to link together the five studies within the project.

GQ1. What is the potential of a microworld-based teaching approach for the development of specific aspects of computational thinking?

The interventions, and their respective programming activities, were designed to promote the acquisition of some key concepts in computational thinking such as sequencing, conditional branching and iteration. Gains in computational thinking were measured using a quasi-experimental pre-test and post-test design in WP2, WP3 and WP4.

GQ2. What is the potential of a microworld-based teaching approach for the development of specific aspects of literacy?

WP2, WP3 and WP4 provided quantitative data by which to test for improvements in clearly defined aspects of literacy development including sentence structures and word families. These aspects of literacy were fully mapped to the Literacy and Numeracy Framework by the Welsh Government (2011).

GQ3. What recommendations can be made for microworld-based classroom practice?

WP1 and WP3 use learning journal/survey data to provide qualitative insights into learner perceptions regarding the microworld designs that were implemented through the interventions.

WP4 and WP5 further extends the qualitative analysis by incorporating two peer talk analyses of paired programming interactions using pre-existing coding frameworks.

1.6 Overview of Findings

This project has contributed new data relating to how extensible block-based programming languages enabling custom block creation can be used to develop specific aspects of literacy and computational thinking. The key findings are summarized here, but for a fuller discussion please see chapter 9.

GQ1. What is the potential of a microworld-based teaching approach for the development of specific aspects of computational thinking?

Independent samples t-tests measuring computational thinking as a whole (not deconstructed into individual computational thinking concepts and practices) reveals statistically significant improvements in mixed-attaining settings following the microworld-based interventions. Cohen's D effect size calculations reveal that the microworld interventions had a moderate impact upon the aspects of computational thinking that were tested for. The ability to sequence a set of programming instructions was improved particularly well, though classroom talk analyses demonstrate benefits far wider than this in areas such as iteration, abstraction and debugging.

GQ2. What is the potential of a microworld-based teaching approach for the development of specific aspects of literacy?

In mixed-attaining settings, the literacy aspects that were tested for showed improvements with a greater than 90% confidence level for change. Further, Cohen's D calculations demonstrated a moderate effect size. The aspect of literacy that saw the greatest improvement was the ability to recognise different poetry forms. Again, however, classroom talk analyses revealed benefits in other areas such as the ability to recognise word classes and draw links between related poetry forms. Further, there is some evidence to posit a correlation between high performance in the literacy and computational thinking aspects, though the weaker confidence level here warrants further research.

GQ3. What recommendations can be made for microworld-based classroom practice?

From the literature review and empirical work, it is possible to provide nine recommendations for educators wishing to incorporate aspects of microworlds-based pedagogy into their own teaching practice.

R1. Language learning is a constructive context for microworld-based learning

R2. A carefully designed *teaching scheme*, with appropriate resources, should increase access to the microworld

R3. Ensure that microworlds provide sufficient challenge as part of mixed-attaining teaching

R4. Use pair programming as a pedagogical technique for working with microworlds

R5. Incorporate a design exhibition format into the microworld-based teaching scheme

R6. Choose a programming platform with a customizable user interface

R7. Plan opportunities for dialogic teaching when working with microworlds

R8. Establish a programme for teacher CPD in microworld-based learning approaches

R9. Consider the opportunities for microworld-based learning in light of school transformation in Wales

2 Review of the Literature

This literature review examines the microworlds approach to learning through programming in education from multiple academic perspectives. The review will address *epistemological* questions about ways of knowing, *sociological* questions about design, and *technical* questions about programming. Rather than choosing to consider each of these perspectives in turn, the review is instead structured thematically across the seven sections that are identified in the introductory overview.

2.1 The Development of Constructionism in Programming Education

The ways in which ICT has been applied in education settings has changed radically. Any attempt to comprehensively recount the diversity of roles played by ICT in education would clearly be flawed, inevitably failing due to the scale of the task. What is needed, at least initially, is a conceptual framework to bring order to this complexity. Robert Taylor's introductory chapter in *The Computer in the School: Tutor, Tool, Tutee* (1980), provides such a framework. This seminal text features in most literature reviews in the domain of educational technology and takes place at a time when ICT was making its debut in schools. It may be read as an historical commentary of past modes of ICT delivery in education.

2.1.1 The Computer as Tutor, Tool, Tutee

Ultimately, what Taylor did was simple: he constructed a trichotomy to shape future thinking about the way in which learners interact with ICT. There are three identified uses of the computer as *tutor*, for computer-aided learning; *tool*, for clerical and routine tasks; and *tutee*, for programming. It is interesting to note the level of humility contained in the work. Taylor does not suggest that all learning with ICT fits into one category without overlaps or uncertainties. Nor does he argue that his framework is definitive and unchanging:

> The point is, one need not be bound by this framework. If modifying it or replacing it by an alternative framework helps [...] such modification or replacement is in order. (Taylor, 1980, p.7)

So as Taylor himself recognises in 1980, there may come a time where a different paradigm may be needed for theorizing the use of ICT in schools. Indeed, Taylor (2003) later reimagined his framework in response to the development of the Internet and pointed to factors such as ease of communication and collaboration. Taylor's (1980) tripartite model is limited in that it focuses solely on the *practice* of educational technologies. More recent models, such as that by Lievrouw and Livingstone (*see* Selwyn, 2011 for an account), have viewed educational technology more holistically and explored the relationship between *artefact* (the device), *practice* (the learning activity) and *context* (educational setting). This maps to the microworlds carried out in this thesis where block-based programming focusing on aspects of literacy (the *practice*) is carried out using a desktop computer (the *artefact*) in a secondary school classroom (the *context*). For now, a closer look at Taylor's three-noun model of tutor, tool and tutee is required. These terms describe distinctly different modes of interaction between the learner and educational technology.

The first of these modes, where the computer takes on the role of the *tutor*, is exemplified most clearly in software packages such as *SuccessMaker*: instructional software that uses on-screen tests to determine a personalized pace of learning (Pearson Education, 2009). In the contemporary, virtual learning environment (VLE) platforms such as Moodle (*see* Moodle Trust, 2013a) enable resources to be uploaded to the web so that the learner may assimilate key points of information. This information is then tested by an automated assessment such as a Moodle quiz, which will inform the learner whether further revision is needed. In Taylor's (1980) words, 'the computer presents some subject material, the student responds, the computer evaluates the response, and, from the results of the evaluation, determines what to present next' (p. 3). The computer tutors the learner and leads them through online tests to assess the level of knowledge acquisition. Taylor's understanding of tutor-mode computing is expressed as follows:

> [T]he computer tutor keeps complete records on each student being tutored; it has at its disposal a wide range of subject detail it can present; and it has an extensive and flexible way to test and then lead the student through the material (*ibid.*).

In contrast to this, the computer may be used to 'preserve intellectual energy by transferring necessary but routine clerical tasks of a tedious, mechanical kind to the computer' (*ibid*.). The learner engages in higher order tasks, such as preparing effective text for a brochure, whilst the computer completes lower order tasks such as maintaining consistent line spacing on a word processor page. The computer, in this example, is being used predominantly as a *tool* for the learner. Tool-mode computing, for Taylor, has useful applications in many areas of school life:

> [T]he tedious recopying of edited manuscripts of texts or even music can be relegated to the computer through word or musical notation processing software; the laborious drawing of numerous intermediate frames for animated cartoons can be turned over to the computer through graphics software (ibid*.,* p.3).

Finally, the computer can take a different role: as the *tutee* rather than the tutor or tool. When the computer is the tutee, the learner takes the role of the programmer – he/she 'talks' to the computer in a programming language it understands. A key example in schools would be the use of a LOGO programming language to learn the core concepts in computer programming as well as in other disciplines such as mathematics. Programming using a computer is an integral skill for

Taylor, serving as a means to 'enable the child to link his or her experience to the deep, fundamental mathematical ideas we most want children to learn' (*ibid.*, p. 7).

Taylor is clear that it is the latter mode, where the learner programs the computer and the computer takes on the role of *tutee*, that confers the greatest educational benefit for the learner. Here Taylor uses teaching as a metaphor for the computational practice of writing a computer program. As we will shall see later, the microworlds-based approach to learning provides a quintessential example of this type of interaction. Arguably, however, it is the first mode of computing that has traditionally dominated the classroom.

2.1.2 Instructionism: The Computer as Tutor

Taylor's first mode of computing, that where the computer assumes the role of tutor, has also been termed the *instructionist* approach to educational technology. In a speech to a conference of educators in Japan, Seymour Papert (ca. 1980) explains this approach:

> Instructionism is the theory that says, "To get better education, we must improve instruction. And if we're going to use computers, we'll make the computers do the instruction." And that leads into the whole idea of computer-aided instruction (*ibid*.).

Later, in The Children's Machine (1993a), Papert uses Freire's criticism of instructionism to illustrate the epistemological connotations of such a model:

> Paulo Freire expresses the criticism most vividly in his description of school as following a "banking model" in which information is deposited in the child's mind like money in a savings account (p. 14).

According to Freire, instructionism views the learner as an antenna to receive transmitted knowledge: the mind is a "vessel to be filled" (*ibid.*). The transmitter of forms of knowledge is the teacher, or in the case of educational technology, the computer. This epistemological model has given rise to software built around the idea of 'drill-and-practice'. *Khan Academy* (2012), for example, is a very successful website that has gained popularity subscribing to this learning paradigm. The site is split into two main sections 'Watch' and 'Practice'. Learners are presented with new forms of knowledge and then participate in computer-based assessments of the knowledge they have learned: a 'learn it'/'test it' model of learning.

Al-Khatib (2009) further explains the instructionist paradigm in terms of a *narration*. The teacher narrates a story of connected facts, and it is the role of the learner to 'learn their lines' in this particular story so that they can recite them at a later date. She argues that this paradigm of learning is a *passive* process for the learner, something that is often implied (quite strongly) by other writers (for example *see* Papert, 1993a), though not always explicitly articulated. The instructionist model, according to Al-Khatib, views the teacher as the *active* agent of knowledge and the learner is a *passive* recipient of it:

> The students' role, in the transmission model, is to accumulate, memorize and reproduce mechanically the narrated content. Education becomes an act of depositing knowledge and learning becomes a *passive* process [*my italics*] where learners receive, memorize and repeat given information instead of getting involved in the learning process (*ibid.*).

Al-Khatib's framing of the instructionist model reads controversially and perhaps overstates the critique of instructionism. One may suggest that he in fact presents an incorrect argument that is premised on a confusion between notions of passivity and agency. Instructionist pedagogy leads learners to actively construct meaning by associating new information with previous knowledge they have already learned. The act of memorizing and repeating information, in this way, is an active not a passive learning process.

For Papert (1993a), however, instructionism proved limited in its application and scope. He writes about his dissatisfaction with the way in which instructionist ideology is embedded in the culture of teaching and learning and permeates throughout language. For Papert, the simple sentence 'the teacher teaches the child' serves as a case in point (p. 83). The teacher is the active subject in this sentence whereas the learner is the passive object. Teaching requires careful attention, with learning happening only when good teaching is put in place. For Papert, this ideology is compounded by a semantic void that exists when attempting to describe methods of learning as opposed to methods of teaching:

> Why is there no word in English for the art of learning? Webster says that the word *pedagogy* means the art of teaching. What is missing is the parallel word for learning (*ibid.*, p. 82).

Papert's candidate for a term to describe the art of learning was *mathetics,* which he derived from the Greek family of words for learning (*ibid.*, p. 84). He attempted to reverse a perceived instructionist hierarchy which privileged teaching over learning and in doing so he constructed the learner as the active agent in the learning process. Papert proposed a revaluation of how we learn by placing the learner, not the teacher, right at the heart of the learning process. One of the greatest influences on Papert's ideas was a theory of learning by Jean Piaget that has become known as *constructivism*.

2.1.3 Piaget and Constructivism

In a textbook for students studying childcare and education, the entry for Piaget summarizes his greatest contributions: '[a] theory that the mind of the child evolves through a series of pre-determined stages to adulthood' (Pound, 2006, p.36). This is an entirely accurate entry, but it nevertheless illustrates a real problem that Papert identified with the reception of Piaget's ideas. In an interview with *Time* magazine, he comments:

> Although every teacher in training memorizes Piaget's four stages of childhood development, the better part of Piaget is less well known, perhaps because schools of education regard it as "too deep" for teachers. Piaget never thought of himself as a child psychologist. His real interest was epistemology - the theory of knowledge - which, like physics before Newton, was considered a branch of philosophy before Piaget came along and made it a science in its own right (Papert, 1999).

For Papert, the most important thing about Piaget's work was not that he identified a path of stages for intellectual development from the sensorimotor through to formal operations. Instead, Papert focused on the epistemological connotations of Piaget's ideas for understanding knowledge formation.

Piaget was a self-termed 'genetic epistemologist' who rejected the instructionist theory of knowing. For Piaget, knowledge was actively constructed by the learner themselves rather than being passively received by the teacher or computer. As Boden (1994) points out, for Piaget 'knowledge is [...] actively constructed, by a dialectical process of assimilation and accommodation' (p. xi). Learners fit external input into their existing mental models through a process of assimilation. In turn, learners change their mental models based on this input through a process of accommodation (*ibid.*).

The processes of assimilation and accommodation operate so that the learner may feel comfortable about their own thoughts in relation to the external world: this is something that Piaget termed *equilibration*. When a young child sees a bird flapping its wings in mid air, she may match the flapping wings in reality to the mental model of flying in her mind. This mental model is called a *schema* (an internalized representation of knowledge), and the process of matching an experience in reality with something already known is called *assimilation*.

Sooner or later, however, the child will see an aeroplane. The child will see that there are no flapping wings but that flying is still happening. She is no longer comfortable with the schema that all flying is caused by flapping wings, and there is a moment of *disequilibration.* In order to feel comfortable with her own thoughts again, and in order to restore *equilibrium*, the schema that all flying is caused by flapping wings must be changed. This process is called *accommodation*: something experiential has caused the learner to change a mental model.

It is important to point out that the cognitive change that follows from the process of accommodation is not always conscious (in the sense that the child may not simply *decide* to change a schema). In the example above, the misconception caused by the 'faulty' schema that all flying works by moving wings up and down may not be changed until several visits to the airport.

Piaget's key point, put simply, is this: learners make knowledge, they do not receive it. This theory of knowing is termed *constructivism* and Piaget is a constructivist. Assimilation and accommodation are merely the mechanisms by which a learner constructs knowledge as they experience the world. The role of experience in the real world, for Piaget, is important because thought is internalized action. Piaget's observations of childhood development led him to conclude that processes such as logic and reasoning are rooted in real-world action, not language (Robson, 2006, p. 14).

In his paper discussing the consequences of constructivism for learning with technology, Dalgarno (2001, p. 184) begins by mapping instructionist and constructivist approaches to developments within the domain of psychology. Dalgarno sees traditional, instructionist approaches as aligned with the *behaviourist* view of learning and its ideology of drill-and-practice. He aligns constructivism, on the other hand, with a *cognitive view* of learning because of its focus on the development of mental models:

> The first development, in the field of psychology, has been the demise of the *behaviourist* view in favour of the *cognitive* view. [Italics in original] A behaviourist view of learning emphasizes teaching strategies that involve repetitive conditioning of learner responses. A cognitive view, on the other hand, places importance on the learner's cognitive activity and the mental models they form (*ibid.*).

When discussing constructivism it is tempting to construct what poststructuralist theorists or postmodernists may call *metanarratives*. What Dalgarno's paper reminds us, however, is that constructivism is by no means a single, coherent and all-encompassing theory of how learning happens. In particular, he identifies three modes of constructivism. First, *endogenous* constructivism emphasizes the individuality of the learner and situates the teacher as a mere facilitator of learning experiences. Second, *exogenous* constructivism is more sympathetic to formal instruction, but only when this is supplemented with plenty of exercises where learners are cognitively active. Third, *dialectical* constructivism purports that scaffolding is required by teachers and emphasizes the important of collaboration with peers (*ibid.*, p. 185).

Several questions are raised by the above distinctions. Will endogenous constructivism lead to an anarchical classroom experience if practiced in reality? Is exogenous constructivism actually constructivism at all? With all of these differences *within* constructivism, is it possible to form one guiding principle that all modes of constructivism share? Delgarno suggests the following as a candidate to answer the latter question: 'learning occurs when the *learner's exploration* [my italics] uncovers an inconsistency between their current knowledge representation and their experience' (*ibid.*, p. 184).

The key principle which binds together all forms of constructivism, then, is a learner-focused theory of knowing that places emphasis on the importance of exploration or Piagetian experience. With this definition in mind, we now turn to a fourth mode of constructivism. It is this fourth mode that underpins the guiding questions of this thesis. It is called *constructionism*.

2.1.4 Papert and Constructionism

Constructionism, as conceived by Papert, extends beyond constructivism to embrace learning through construction of external models:

> Constructionism - the n word as opposed to the v word - shares constructivism's connotation of learning as 'building knowledge structures' irrespective of the circumstances of the learning. It then adds the idea that this happens especially felicitously in a context where the learner is consciously engaged in constructing a public entity, whether it's a sandcastle on the beach or a theory of the universe (Papert and Harel, 1991 p. 1).
>
> The word constructionism is a mnemonic for two aspects of the theory of science education underlying this project. From constructivist theories of psychology we take a view of learning as a reconstruction rather than as a transmission of knowledge. Then we extend the idea of manipulative materials to the idea that learning is most effective when part of an activity the learner experiences is constructing a meaningful product (Papert, 2008a).

Constructivism is a process which involves learners building their own mental models. Constructionism, however, takes this further by suggesting that this process of making mental models is at its most powerful when the learner is making something in the real world. Put simply, constructivism says that learners *make mental models*. Constructionism, on the other hand, extends this by positing that learners *make mental models* through making *physical models* in reality. Constructionism, in this way, involves two senses of making: making within the *mind* and making within *reality*. Resnick and Kafai (1996) explain this constructivism-constructionism distinction clearly in their introduction to the edited volume *Constructionism in Practice*:

> [Constructivism suggests that] children don't *get* ideas: they *make* ideas. Moreover, constructionism suggests that learners are particularly likely to make new ideas when they are actively engaged in making some type of external artifact - be it a robot, a poem, a sand castle, or a computer program - which they can reflect upon and share with others (p. 1).

Lessig (2004) recalls the Chief Scientist of the Xerox Corporation - John Seely Brown - to explain this idea (p. 45). Brown believes that learning takes place through a process of *tinkering*. As Lessig explains, this frames a mode of learning where abstract ideas are formed through concrete experiences. The central premise of constructionism may be summarized as this: learners construct their *knowledge* through their constructions within the external world. Learners make *mental* models by making models *in reality*.

Extending beyond this dual sense of 'making' are three further constructionist perspectives that need to be considered. Resnick and Kafai (*ibid.*) use these three perspectives to structure their edited book on the subject. Each of these themes will now be considered in turn.

2.1.5 Themes in Constructionism I: Learning by Design

Kafai (1996a) puts forward the suggestion that design plays a vital role within constructionist learning approaches. She identifies two different types of design: *professional design* is concerned with the final outcome and *learning design* is concerned with the process leading up to the final outcome. For Kafai, it is the latter that contributes most greatly to the learning process: 'learning through design is not exclusively represented in the final product, but also in the process of doing it' (p.73). It is the learner's design process, in other words, that determines the effectiveness of learning: it is not how effective something looks *after* it has been completed.

Constructionism is very often referred to as a 'learning-by-making' theory of knowing. Indeed, Papert and Harel (1999) open their edited volume *Constructionism* by at first recognizing, and then critiquing, the 'simple catchy version' of constructionism as 'learning-by-making' (p. 1). Papert and Harel argued that constructionism was much more than this: they called for an understanding which was 'much richer and much more multifaceted [...] than could be conveyed by any such formula' (*ibid.*). Whereas Papert and Harel disliked the phrase 'learning-by-making' because of its oversimplification, one should also be cautious about its use for a different reason. The phrase 'learning-by-design' is a more accurate, succinct formulation of the central principle of constructionism.

> Constructionist theory suggests a strong connection between design and learning: It asserts that activities involving, making, building, or programming - in short, designing - provide a rich context to learning (*ibid.*, p. 4).

The argument suggested here is an important semantic one: the term 'design' illustrates a key constructionist theme much more accurately than 'making'. A pocket dictionary defines *make* as a verb meaning 'cause to exist' and *design* as a verb meaning 'plan, intend' (Oxford English Minidictionary, 1995, p. 311 and p. 137). It is perfectly possible, if one subscribes to these operational definitions, to *make* something without thinking about it. According to such a model of making, a group of learners could be given identical pieces of paper and identical sets of instructions about where to place the folds in order to produce a paper aeroplane. This *is not* constructionism. Design, on the other hand, additionally requires the learner's intention and planning: the learner needs to *think* about what they are making rather than merely carrying out the teacher's instructions. This *is* constructionism.

In his introduction to the second edition of *Mindstorms*, Papert (1993b) echoed such concerns though he framed them differently. The coding examples provided in the first edition of his book were often given to learners as exercises or tasks to complete in the LOGO programming language. He calls for more emphasis to be given to personally meaningful design when adopting a constructionist learning approach:

> In Mindstorms I introduce Logo programming through examples such as writing tiny programs to make a square and a triangle and putting them together to form a house. Unfortunately, the conditions of work in the school often mean that what was described in the book as a tiny first step was all that could be done (p. xvi).

2.1.6 Themes in Constructionism II: Multiple Ways of Knowing

A second thread that runs through constructionism has been termed *epistemological pluralism*. This accepts that there is more than one way of knowing, thinking about and learning something. Traditional discourses of epistemology privilege abstract thought over alternative ways of thinking. The Computing at School (CAS) group, for instance, sets out a possible road map for computational thinking development in primary schools that begins with decomposition of simple problems in KS1 before increasing abstraction is encountered in KS2 (CAS, 2016). *Epistemological pluralism* problematizes such discourses of epistemology as Turkle and Papert (1991) explain:

> For some people, what is exciting about computers is working within a rule-driven system that can be mastered in a top-down, divide-and-conquer way. Their structured 'planner's' approach [...] is validated by industry and the academy. It decrees that the 'right way' to solve a programming problem is to dissect it into separate parts and design a set of modular solutions that will fit the parts into an intended whole. [Some programmers] are not drawn to structured programming; their work at the computer is marked by a desire to play with the elements of the program, to move them around almost as though they were material elements - the words in a sentence, the notes on a keyboard, the elements of a collage (p. 136).

The Piagetian, constructivist approach to intellectual development privileges abstract thought over other modes of thinking in the sense that abstract thought is seen as a more developed stage. Piagetian learning here reflects the normative cultural construction of knowledge acquisition. Formal reasoning is achieved when a child reaches the final point of their journey through the prior stages of intellectual development. Abstract thought is placed at the top of the hierarchy of epistemology: it is somehow seen as more 'advanced' than other types of thinking. For Turkle and Papert, this hierarchy of value is present throughout both industrial and academic institutions (*ibid.*).

Constructionism, on the other hand, views abstract thought as a *way of knowing* and not as a *stage* of intellectual development. Unlike constructivism, constructionism does not view abstract thought as more developed than other modes of thinking. Some learners may be 'planners' or 'top-down' thinkers that prefer to create programs in a structured and modular way. That is perhaps illustrated by Ethan in WP5 (*see* section 8.3.2.iii) who is primarily concerned with the construction of the block kit and how to create his own custom blocks. Such a way of knowing may be aligned with the traditional *dominant* positioning of the *abstract*.

Other learners, however, may adopt a 'bricoleur-style' or 'bottom-up' approach that is less-structured and more playful. This learner profile is reflected in the data collected here by looking at Ovyaa in WP5 (*see* section 8.3.2.i) who was much more concerned with finding and populating detailed word lists before the process of generating poetry began. This way of knowing may be aligned with the traditional *subordinate* positioning of the *concrete.*

Constructionists do not privilege either approach and attempt to encourage a co-presence of ways of knowing in all learning situations. As Kafai (1996a) explains when implementing a game design research project:

> These two views ['planner' and 'bricoleur'] suggest that students may approach the design task from different ends [...] and think about it in different ways depending on their personal preferences. My intention in the game design project was to offer a design activity that provided multiple avenues for students to approach a complex problem (p. 74).

2.1.7 Themes in Constructionism III: Reimagining Learning Communities

A third theme of constructionism is its reimagining of learning communities or, as Papert (1993b) puts it, 'images of the learning society' (p. 177). In *Mindstorms*, Papert questions some of the fundamental assumptions that are made about learning. Papert's questioning is not limited to the ontology of learning: he further problematizes its institutionalization and the effectiveness of the Western 'school' system.

Papert's fascination with Brazilian samba schools (*ibid.*, p. 178) illustrates a very different community of learning to the one that can be seen in most classrooms. Indeed, if a constructionist like Papert were to design a learning environment from scratch, it might look very different to a 'school' as we know it. For Papert, the samba school model possesses a number of key characteristics that a learning environment should have:

> The Samba School has a purpose, and learning is integrated in the school for this purpose. Novice is not separated from expert, and the experts are also learning (*ibid.*, p. 179).

Papert's first point is that the idea of the Samba School firmly embeds learning within experience. Learning is not composed of hypothetical maths problems or abstract algebraic formulae: learning is deeply embedded within the everyday, in this case the performances that will take place at the Rio de Janeiro carnival. The Samba School model presents a process-oriented approach where all learners share communal goals within an active and participatory culture.

Second, Papert uses the Samba School model to deconstruct the perceived binary dichotomy of teacher-learner that has shaped Western discourse on education. In the Samba School model, experts are learning whilst novices are teaching other novices. The Samba School teacher, as a member of the learning community him/herself, acts as a facilitator of learning but never delivers any kind of 'curriculum' that has been planned in advance (*ibid.).*

It is appropriate at this point to think critically about the notion of the Samba School as a general model of learning. The learning experience with such a model is limited to the experience (and associated language and practices) gained from participation in a specific closed community. There is a strong case, then, for a more planned curriculum to ensure that learners achieve varied and balanced exposure to skills and concepts across different subject domains. It is also untrue to suggest that professional educators in Western schooling never assume the role of learner. Teachers in a Welsh secondary school will regularly undertake CPD to improve both their subject and pedagogical knowledge, though the Samba School model of learning is different because they work for the same ends, for the same purpose, as the novices.

Papert's observations with the Brazilian samba schools could be said to be idealized (*see* Zagal and Bruckman, 2005, p. 88, for example) and difficult to implement in practice. His observations resonate in the present day, however, with such activity networks as the Intel Computer Clubhouse. This is a network of 'drop in centres' in the US where learners from socioeconomically-disadvantaged backgrounds are able to explore creative applications of computing in informal learning settings (*ibid.*).

Constructionism, then, makes clear statements regarding the role of (i) design, (ii) multiple ways of knowing and (iii) community in learning experiences. We have seen how the elements of Papertian constructionism are premised on the foundations of Piagetian constructivism. But what is interesting about the latter theme, Papert's focus on the Samba School learning community, is that this seems to signal a departure from the Piagetian model of learning.

2.1.8 Piaget versus Vygotsky: The Competing Roles of the Individual and the Social

Piagetian learning is a form of constructivism because it states that knowledge is formed by the individual experiences of the learner. To re-visit Dalgarno's (2001) tripartite model of constructivism that was discussed earlier, Piagetian learning employs an *endogenous* mode of constructivism. Piaget was curious about how the infant developed into a thoughtful, imaginative adult through a change that was part biological and part cognitive.

The cognitive dimension to this involved the refinement of schemata through encountering moments of disequilibriation in lived experience. Piagetian learning views knowledge construction as an internal process achieved through the mechanisms of assimilation and accommodation. As learners explore their environment, they will be faced with a number of inconsistencies between their schemata and the external world. In order to address this inconsistency, learners will attempt to balance the process of fitting external stimuli into their existing schemata (assimilation) with the process of changing the schemata to enable external stimuli to fit (accommodation).

The key point to note here is that this account of knowledge construction is rooted in *individual*, not *social*, experience. Piagetian learning resides in the interaction of an *individual learner's* internal mental models with their own external real-world experiences.

Vygotsky, on the other hand, was a psychologist who famously devised a theory of the *zone of proximal development* (ZPD). ZPD, according to Bruner (1982), is 'the child's ability to recognize the value of hinges and props even before he is conscious of their full significance' (p. 852). The ZPD is the difference between what a learner *can* do *independently* and what a learner has the *potential* to do with guidance from *others*. The social aspect to learning is referred to by Vygotsky (1978) himself in an often-quoted passage defining the ZPD:

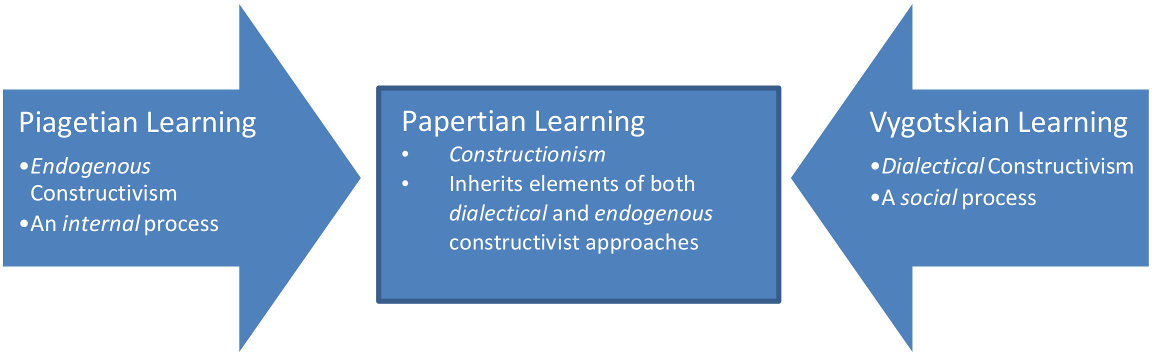
> [T]he distance between the actual development level as determined by independent problem solving and the level of potential development as determined through problem solving […] in collaboration with more capable peers (p. 86).

Papert's interest in the role of the Samba School community extends beyond personal experience by showing an interest in the social mechanisms of learning. In this way, it is possible to draw links between Papertian constructionism and the ideas of Vygotsky. Vygotsky popularized what has become known as a social or *dialectical* mode of constructivism. Dialectical constructivism shares with the endogenous mode the idea that knowledge is constructed by the learner. What dialectical constructivism adds to this is that knowledge construction is reliant on effective *social scaffolding* (*see* Wood, Bruner and Ross, 1976).

For Vygotsky, it is the role of the classroom teacher to locate the learner's zone of proximal development and plan for the correct amount guidance to be administered by themselves, the learner's peers and support staff. This guidance must be well-matched to the learner's current capability. Finding the ZPD involves the teacher finding the correct balance between progression and challenge, but - as with Piagetian approaches – there is an inherent difficulty in establishing what 'stage' or 'zone' an individual is in.

Vygotsky did not use the term scaffolding *per se*, but he did suggest that learners should be provided with an appropriate amount of guidance to carry out a task initially. It is important to note that the teacher is not the sole source of scaffolding for Vygotsky: guidance could come from other sources such as peers in group-work activities.

The concept of differentiation in the classroom, a concept with which all teacher readers are undoubtedly familiar, links in with this notion. The teacher should exercise his/her judgment to remove initial scaffolding at an appropriate point in the learner's development to enable them to carry out the task with independence. A further link to an instructional design context is provided by Dillenbourg (2013) and his work on the orchestration of learning experiences: 'orchestration refers to how a teacher manages, in real time, multi-layered activities in a multi-constraints context' (p. 485).



Piagetian Learning

- *Endogenous* Constructivism
- An *internal* process

Papertian Learning

- *Constructionism*
- Inherits elements of both *dialectical* and *endogenous* constructivist approaches

Vygotskian Learning

- *Dialectical* Constructivism
- A *social* process

Figure 1: Positioning a Papertian Learning Approach

Papert's focus on the teacher-learner relationship connects with the Vygotskian perspective of the impact of others on learning. Vygotsky wrote that the mechanism of scaffolding needed to be well-matched to the individual learner's zone of proximal development. Papert, on the other hand, wrote that the learner should take control of what is learned but with teacher guidance. Shared by both writers is the prominence given to others in providing guidance throughout the learning process. The notion of social interaction that is so important for Vygotsky is also important for Papert, as the quotations from both writers below suggest:

> The scientific concepts evolve under the conditions of systematic *cooperation* [my italics] between the child and the teacher (Vygotsky, 1962, p. 148).
>
> The [...] teacher will answer questions, *provide help* [my italics] if asked, sometimes sit down next to a student and say 'Let me show you something.' What is shown is not dictated by a set syllabus (Papert, 1980a, p.179).

A Papertian Constructionist paradigm, then, inherits elements of both Piagetian and Vygotskian learning approaches. Piagetian learning is a process defined by the *internal* mechanism of equilibration whereas the Vygotskian zone of proximal development emphasizes the importance of *social* scaffolding. Papertian Constructionism provides a bridge between the two poles of endogenous and dialectical constructivist approaches, as shown diagrammatically in *figure 1*.

2.1.9 Teaching and Learning with Microworlds

According to Papert, the role of the teacher changes when learning with microworlds. The focus is not on improving teaching to maximize the receipt of knowledge (as instructionism would have it). Instead, it is about setting up the right conditions for learning: creating an environment for exploring, creating, doing. In Papert's own words, the constructionist teacher 'giv[es] children good things *to do* [italics in original] so that they can learn by doing much better than they could before' (ca. 1980).

2.1.9.i The Role of the Teacher

Papert's construction of the role of the teacher gives emphasis to the *planning* of learning experiences through effective microworld design. He falls short, however, of providing a comprehensive account of the role of the teacher *throughout* the learning process. Jonassen (1999) addresses this question through a framework for supporting learning within web-based constructionist learning environments. The framework suggests that there are three mechanisms of instructional support exercised by the constructionist teacher: modelling, scaffolding and coaching. It is important to note that though Jonassen focuses on scaffolding provided by the teacher, resources and other contextual materials are a key part of learning with microworlds. WP5, for instance, used a series of online tutorials and unplugged learning materials in addition to teacher input. Each of Jonassen's support mechanisms now be considered in turn to develop an understanding of the role of the teacher within the microworld-based learning approach.

The first characteristic of modelling refers to the way in which the teacher illustrates expected performance in order to assist the learner with their own exploratory actions. Carefully timed whole-class demonstrations should model completed programs in order to assist learners with the development of their own programming creations. In WP2 (*see* appendix D), a completed example of the poem generator microworld enabled learners to see the individual steps that are needed to be taken in order to achieve a successful outcome: in this case a haiku. This is important because 'worked examples enhance the development of problem schemas and the recognition of different types of problems based on them' (*ibid.,* p. 8).

Second, when learning within a microworld, it is the role of the teacher to provide appropriate scaffolding throughout the learning process. Scaffolding is necessary in order to 'provide temporary frameworks to support learning and student performance beyond their capacities' (*ibid.*, p. 9). Appropriate scaffolding involves 'adjust[ing] the difficulty of the task to accommodate the learner' *(ibid.*). A method used in WP5 were the staged tutorials (*see* appendix G) that incrementally introduced new microworld primitives to learners as they were ready to progress. A further example in WP4 is the series of 'unplugged' activities (*see* appendix G*)* that were designed to provide initial support prior to the computer-based activities.

Finally, the constructionist teacher should make carefully-timed coaching interventions throughout the learning process in order to advance progress of individual learners or small groups of learners:

> A good coach motivates learners, analyzes their performance, provides feedback and advice on the performance and how to learn about how to perform, and provokes reflection and articulation of what was learned (*ibid.* p. 9)

The classroom dialogue analysis in WP5 contains several examples of where coaching interactions from by the teacher led to learning gains. In CE8 (*see* table 31) for instance, careful coaching from the teacher regarding a specific programming block helped Ethan to correctly debug an issue that was causing his poems to not be displayed. CE5 *(see* table 28) provides a further example where coaching from the teacher helps Ethan in the construction of a block variable.

Fundamental to the learning that took place in both of these cases were the exchanges of dialogue that took place between the teacher and the learner. The importance of classroom dialogue when learning with a microworld is now discussed further.

2.1.9.ii Classroom Talk

Monologic classroom talk describes a classroom where the goals of classroom dialogue are tightly controlled by the teacher (*see* Lyle, 2008). An example of monologic classroom talk may be teacher-led diagnostic questioning where the desired responses by the learner are planned in advance. In the examples above, however, the dialogue is initiated by the learner as a result of real issues they experience when exploring within the microworld. This departure from classroom dialogue that is controlled by the teacher is characteristically *dialogic* because the learner is setting the topic for discussion in order to create new knowledge (*see* Alexander, 2006). Dialogic classrooms encourage the learner to pose questions and take ownership over steering the dialogue that takes place:

> Dialogic teaching is distinct from the question-answer-tell routines of so-called 'interactive' teaching, aiming to be more consistently searching and more genuinely reciprocal and cumulative (*ibid.*, p. 1)

The latter example from WP5 above demonstrates how, in the dialogic classroom, learner-led conversation with the teacher played a key part in the learner finding out how to construct a block variable. It is important to note that a link can be made here between the primacy of social interaction within dialectical constructivism (*see* Dalgarno, 2001) and learner-led classroom talk within dialogic teaching.

A further similarity between the approaches is the importance of peer interaction in addition to teacher-learner interaction. Again in WP5, CE11 (*see* table 34) shows an excerpt of dialogue that takes place between experienced attendee at the coding club Rokas and his less-experienced friend James. Rokas not only provides reassurance to his friend James, but he also coaches him on the computational thinking practice of reusing and remixing.

The exchange above takes place as part of a pair programming approach that was used in WP5. Plonka et al. (2011) explain the key roles assumed by programmers as part of this approach:

> There are two roles in each pair: driver and navigator. The programmer who is typing is typically called the "driver" and the other one is called the "navigator" [...] The navigator […] is responsible for reviewing the driver's work […] whereas the driver is responsible "for typing the code" (p.43).

In the example in CE11, Rokas is the navigator in the interaction. He explains to James, the driver in the interaction, that he should reuse and remix code from a different session where Rokas was the driver. James, meanwhile, follows the advice of the navigator in order to start developing his own programming code for making poetry.

The potential gains in learning derived from a pair programming are well documented. Potential benefits include a greater enjoyment of the coding progress in addition to a faster production of coding solutions. Cockburn and Williams' (2000) literature review on the subject lists some of the benefits:

> - The team solves problems faster […]
> - The people learn significantly more […]
> - People enjoy their work more (p. 9)

Further, there is evidence to suggest that programming in pairs enables a greater understanding of the programming process across the pair. It is possible, in other words, for the knowledge demonstrated by the navigator to transfer to the driver through the mechanism of peer discussion during pair programming. As Chong et al. (2005) put it:

> Positive results with pair programming have led to speculation that a collateral benefit of the practice may include […] project knowledge shared efficiently across the development team (p. 43).

To summarise, the teacher in microworlds-based learning plays an active role throughout the learning process. They will model performance outcomes using whole-class discussion, scaffold activities for lower attaining learners, and providing coaching when learners require it.

Further, classroom talk is an integral part of the learning that takes place with microworlds. Dialogic classrooms enable learners to construct new knowledge by taking control over discussing issues with others. The ‘other’ here is not *just* the teacher: pair programming serves as a useful tool for encouraging classroom dialogue and there is growing evidence to suggest that programming in pairs holds strong educational value.

The role of the microworlds-based teacher is multifaceted. Effective *pedagogic* design throughout the learning process is equally as important as the *technical* design during the planning stage of developing a microworld. In *Teaching as a Design Science*, a quote by Diana Laurillard (2012) captures the inherent difficulty of this task:

> The arrival of digital technology over the past three decades, increasingly impacting on work, leisure, and learning, has been a shock to the educational system that it has yet to absorb […] Teaching is not a rocket science: it is much, much harder than that. Rocket science is about moving atoms from a to b; teaching is about moving minds (pp. 2-5).

2.1 Papert, LOGO and the Advent of Turtle Graphics

This section of the literature review begins with a series of very important questions. What is a microworld? How does one know a microworld when one sees one? How does a microworld fit into the paradigm of constructionism that has been discussed so far? Once again, Papert provides a useful starting point for answering these key questions. In the 1960s, Papert and his team at MIT set about creating an educational programming language called LOGO and within this they implemented a microworld called Turtle Graphics. It is now time to look more closely at this microworld.

2.2.1 Introducing the Turtle Graphics Microworld

> The Turtle World was a microworld, a ‘place,’ a ‘province of Mathland,’ where certain kinds of mathematical thinking could hatch and grow with particular ease. The

> microworld was an incubator [...] The design of the microworld makes it a "growing place" for a specific species of powerful ideas or intellectual structures (Papert, 1980a, p. 125).

Papert's choice of words here is very important. He uses words like *hatch*, *grow*, and *species*. This is because, for Papert, the microworld provided a means for learners to acquire knowledge in a *natural* way. An example of natural learning commonly used by Papert and other constructionists is early language development in infants which does not apparently require didactic instruction: language develops naturally through immersion within a linguistic environment. It is worth noting that this is not entirely true since there is an inherent didacticism in early language learning. The idea of immersion in an active and interactive environment is the most important feature, though this rests on underlying developmentalist ideas about natural learning. A microworld, for Papert, is an environment within which ideas in many domains can develop in a similar, natural way.

Papert's choice of words in the previous quotation is important for another reason. Words like *grow* and *hatch* are mixed in with words like *place* and, significantly, phrases like '*province of Mathland'.* This lexicon of place implies both immersion but also a sense of borders or boundaries, which are a key part of a microworld-based approach to learning. Put simply, microworlds should not be reduced to virtual sandpits that practice an 'anything-goes' approach to learning. Microworlds are designed with boundaries and constraints to enable the discovery of key ideas within particular domains of knowledge. As Mitchel Resnick (1997) puts it, 'Microworlds are simplified worlds, specially designed to highlight (and make accessible) particular concepts and particular ways of thinking' (p. 50). Turtle Graphics for example, a subset (or province) of Mathland, may be regarded as a paradigmatic microworld where mathematical ideas could develop in a more 'natural' manner, in much the same way as early language learning in the pre-school infant but with well-defined boundaries and constraints.

More recently, Selwyn (2011) established a link between the Papertian microworld and its roots in constructionism: 'learners build things and naturally encounter problems that require creative solutions' (p. 75). It is important to clarify here that creating the microworld is often distinct from the exploration that takes place within it. In WP5 of this project, for instance, the Poem Generator custom blocks were created by the teacher in Snap! whilst learners explored using these blocks to create poetry. This distinction was somewhat blurred, however, where some learners altered the design of the microworld itself through the creation of their own custom blocks.

2.2.2 The Significance of Playing Turtle

> [A]t the root of it all [Turtle Graphics] is the Turtle. The Turtle must be rated surely as one of the great inventions of educational technology in the latter half of the 20th century (Longman, 2011).

The single most important feature of Papert's Turtle Graphics microworld, as the quotation by Longman suggests, is the Turtle itself. The Turtle can initially take the form of a small pen-carrying machine that moves about the floor by being issued with a sequence of simple commands like 'forward' and 'left'. When placed on a large sheet of paper, the learner can create a series of 2D drawings.

By entering the directional command RT90, the learner will find that the Turtle rotates clockwise around its centre through an angle of 90 degrees. LT90 will see the Turtle rotate anti-clockwise in a similar fashion. A Turtle can also move forwards and backwards with commands like FD10 and BD10. In this example, FD and BD indicates forwards and backwards whilst the number afterwards indicates the number of Turtle steps the Turtle should take in the specified direction. Inherent within the design of the microworld, then, is an Euclidean geometry that enables mathematical exploration on a 2D plane.

Attached to the 'shell' of the LOGO Turtle is a pen which can be lifted up (PU) or placed on the floor (PD). When the turtle lifts his pen up, no trail will be made. When the turtle places his pen down, all of his movements leave a visible trail. Using this simple set of basic commands, the learner is able to direct the turtle to make all manner of creative graphical creations.

The LOGO Turtle Graphics microworld takes this concept of a physical turtle in the real-world and transfers it to a computer screen, as Boyle (2004) explains:

> The […] turtle is initially a small robot that moves across the floor and is controlled from a computer. This transitional object enables the child to make sense of the task in everyday terms and facilitates moving into the world of the abstract. […] Later, and more abstractly, the turtle becomes a cursor on a computer screen (p. 107).

Boyle's quotation here is interesting because it describes the Turtle as a *transitional object*. Transitional object theory is most widely associated with the work of psychoanalyst Donald Winnicott. The phrase refers to the way in which a physical object, such a 'comfort blanket', may assume the role of a mother-substitute in the early stages of child development. This is explained by Turkle (2007):

> Winnicott writes that the transitional object mediates between the child's sense of connection to the body of the mother and a growing recognition that he or she is a separate being. […] Transitional objects let us take things in stages (p. 314).

For Papert, the Turtle assumes the role of a transitional object for the domain of Euclidian geometry. He thought that the development of abstract mathematical ideas must be linked to the Turtle at first until sufficient abstract mathematical knowledge has been acquired, as he explains:

> In order to internalize the mathematical notion the child needs to represent it as (assimilate it to) something already present (the transitional object) in his internal, mental world [...] once a sufficiently rich collection of formal objects have been internalized, these act as internalizers of new ones. Mathematics becomes a self-reproducing system (Papert, 1980e, p. 205).

What is particularly special about the Turtle device, then, is that it is *syntonic* to learner experience. This means that it is possible to ascribe human-like or 'me-like' behaviours to the Turtle since we are able to do all the things in reality that a Turtle can do on screen. Just like a Turtle, we can turn to the right, turn to the left, walk forwards and (though it is not recommended) walk backwards. If a young child finds a part of the programming process difficult, she is able to 'play Turtle' by moving her body 'as the Turtle on the screen must move in order to make the desired pattern' (Papert, 1980a, p. 58).

Playing turtle, for Papert, represents a *syntonic enactment game*. In *Teaching Children Thinking*, he (1980b) explains this phrase further:

> Using an important heuristic we encourage the child to study himself in the situation, and try to simulate his own behavior [...] The child can think of the program as a very simple formal model of himself (pp. 170 - 171).

Syntonicity refers to the way in which the Turtle device embeds learning within the learner's experience. Turtle geometry can be made *body syntonic* because the Turtle moves left, right, forward and back in a way that the learner can act out with their bodies. It is *ego syntonic* since it has an end goal following a sequence of smaller goals or steps, just as the learner does. In this way, the Turtle device embodies a dual sense of syntonicity and is directly relatable to the learner's own life experiences (*ibid.*, p. 63).

Syntonicity is an integral part of the learning process for Papert and it is a term that he defines most simply as 'it goes together with' (Papert, 1980d, p. 202). For Papert, learning is least effective when the content is something that is disassociated from the learner's reality. If learning should connect with the experiences of the learner, it follows that learning should not progress along a rigidly-structured curriculum designed by the teacher. For Papert, any attempt at a pre-determined curriculum of study might lead to dissociated learning since it does not align with the personal experiences of the learner.

As as a constructionist, Papert thought that learning is best achieved through discovery that is led by the learner themselves. This enables a mode of learning that they are able to connect with syntonically, as Papert explains:

> [Children acquire] a vast amount [...] of knowledge before they ever come near the classroom. And this is not knowledge which is acquired in a disassociated way. So I would suggest that syntonic learning may be another good name for Piagetian learning, which is a kind of learning that happens without being taught, without a curriculum (Papert, 1980d, p. 202).

Papert's view here is contentious and is perhaps once again, as Zagal and Bruckman (2005) assert, an example of Papert's idealization of education. More specific critiques of Papert's claims are examined later in the review (*see* section 2.2.4).

2.2.3 The Mathetics of Turtle Graphics

What, then, characterizes the mathetics of how learning happens with Turtle Graphics? For Papert, the LOGO microworld provides a way for learners to engage with a new and exciting 'learnable and lovable mathematics.' The LOGO microworld provided a platform from which to explore a 'creative mathematics' that Papert (1980c) called 'Turtle geometry' (pp. 178 - 181).

The true value of Turtle Graphics, for Papert, extended much further than a mere repackaging of one subject discipline. Papert thought that the learner is able to reflect upon their own thinking processes and take on the role of self-reflexive epistemologist whilst developing their programming creations. They are able, in Papert's (1980a) words, to be 'brought into touch with some powerful general ideas' and acquire more general 'mathetic principles' (p. 60). Papert thought that Turtle Graphics enabled important discoveries about learning processes to be made. Turtle Graphics, he controversially claimed, conferred educational benefits beyond the activity of programming in LOGO.

One of these discoveries, as we have previously seen, is the ability to play Turtle. This is the idea that using familiar knowledge can help with the process of acquiring new knowledge: using the known to solve the unknown. A further important mathetic principle is the notion of *debugging*.

In *Mindstorms*, Papert (1993b) observed discordance in the learning culture of the school-age education system when compared to the learning culture within the LOGO Turtle Graphics microworld. Papert observed a 'right or wrong' culture within the education system of the time. There is a parallel to be drawn between Papert's observations and the focus on 21st Century Learning by the Organisation for Economic Co-operation and Development (OECD) (2008). The OECD critique a stereotype construct of schooling that is defined by binary right vs. wrong answers. Instead, they suggest that schools should cover ‘a very wide range of approaches to learning’ (p. 10).

Papert thought that from the moment they begin school, learners internalize a cultural norm which is endemic within the school system: ‘you either know something, or you don’t.’ (p. 23) Programming, on the other hand, is characterized by an entirely different engineering model of learning. As Papert explains:

> [W]hen you learn to program a computer you almost never get it right the first time. Learning to be a master programmer is learning to become highly skilled at isolating and correcting ‘bugs’ (*ibid.*).

For Papert, Turtle Graphics provided learners with a more productive way of thinking about their own learning. He thought that reflecting on something and thinking ‘I’ve got it’ or ‘I’ve got it wrong’ is an entirely wasted endeavour when learning to program. Instead, learners must reflect upon ways to improve their own process of debugging if they are to become a successful programmer. The activity of debugging is a vital part of learning to program:

> Typically in math class, a child’s reaction to a wrong answer is to try to *forget* [italics in original] as fast as possible. But in the LOGO environment, the child is not criticized for an error in drawing. The process of debugging is a normal part of the process of understanding a program (*ibid.*, p. 61).

Part of what made the Turtle Graphics microworld so important to Papert was the possibility that its mathetic principles could extend beyond the activity of programming in a LOGO environment. He thought that the mathetic of debugging, and indeed of playing Turtle, could extend beyond Turtle Graphics to the experience of learning *in general*. By thinking about their own thinking when programming the Turtle, the learner is able to discover the importance of debugging to the activity of programming. And whilst they can improve their own debugging processes when creating computer programs, learners can also improve their debugging processes more widely in all areas of learning.

> [T]hinking about learning by analogy with developing a program is a powerful and accessible way to get started on becoming more articulate about one’s debugging strategies and more deliberate about improving them (*ibid.,* p. 61).

To explain a further mathetic principle that may be nurtured by the act of computer programming, Papert reverses a dystopian argument made by critics regarding the impact of computation on society. A machine-like mode of cognition is one in where the learner follows an exacting and step-by-step set of instructions, without diverting from a described sequence. For Papert, such a cognitive style does not present cause for concern. Indeed, he claims that thinking like a computer is an important heuristic to acquire when it comes to exhausting possible solutions through the process of solving problems. Papert (1980a) claims that learning to think mechanically like a computer is a beneficial cognitive style to acquire:

> The critic is afraid that children will adopt the computer as model and eventually come to 'think mechanically' themselves. Following the opposite tack, I have invented ways to take educational advantage of the opportunities to master the art of *deliberately* [italics in original] thinking like a computer (p. 27).

Educationalists have expressed concern that increased interaction with computers will lead young learners to develop a 'machine-like' mode of thinking. Future employment, generally speaking, may also be at stake as a result of the increase in automation and changing work patterns. On the other hand, such threats may serve to justify the necessity of computational thinking in education. By learning to think computationally, learners may be better equipped with the thinking skills needed in order to make sense of computerized working environments.

A final mathetic that Papert (1980c) thought was nurtured by Turtle Graphics is modularization. A characteristic of the LOGO environment is extensibility through the definition of custom procedures. By typing the word TO followed by a procedure name and a sequence of commands, the learner is able to tell the computer to remember a sequence of instructions and call these instructions whenever a particular word - the name of the procedure - is keyed in (p. 182). The following example creates a procedure called SQUARE. Now, whenever the command SQUARE is typed, the computer will know how to draw a square with the on-screen turtle.

```
TO SQUARE
FD 10
RT 90
FD 10
RT 90
FD 10
RT 90
FD 10
END
```

After the learner has 'taught' the computer how to draw a square, she can then repeat this process to teach it how to draw a triangle. With a little bit of debugging, the learner can proceed to tell the computer how to draw a house by combining the two things learned earlier: how to draw a triangle and how to draw a square. In programming terms, these user extensions are treated in the same way as the primitive operations.

The process of making procedures that Papert describes here maps to *algorithmic thinking* in the contemporary Computing at School framework for computational thinking (*see* Curzon et al., 2014). He also describes the process of splitting up a problem into its different components. This description maps to the CAS description of *decomposition* (*ibid*.).

Papert (1980a) refers to both practices using the term *modularity*. Learners are able to establish the principle of hierarchical organization (p. 60). The house procedure is made up of two sub-procedures of square and triangle, which are in turn comprised of the primitives RT and FD. For Papert, splitting up a problem into a hierarchy of components in this way is a powerful idea that learners are able to discover in a variety of contexts ranging from the organisation of species to the colour spectrum.

2.2.4 Problematizing Papert

In *2.2.3*, it was seen that Papert makes some very strong claims about the educational benefits of using Turtle Graphics. Equally as strong is idea that the microworld provides an ideal constructionist learning environment. Through programming in this environment, Papert thought that learners gain an understanding of the mathetic principles of playing Turtle, debugging, thinking, modularity.

Perhaps even more boldly, Papert suggests that Turtle Graphics has the potential to develop these learning processes *beyond* the computer lab, outside the original context of programming in LOGO. Turtle Graphics, for Papert, has the power to change learning across all disciplines. These significant claims have, unsurprisingly, been subjected to much challenge.

One of the major claims made by Papert is that programming with Turtle Graphics enables learners to reflect upon their own thinking and problem solving processes. Learners are able to identify the act of debugging when undertaking programming activities, for example, and then extend this mathetic to other areas of learning. Such a claim, however, rests on the assumption that learners are able to exercise conscious reflexivity with regards to their own metacognitive processes. This raises the question of whether or not learners are able to reflect on their own thinking in this way.

A study by Tanner et al. (2011) was carried out in a Welsh Foundation Phase classroom to determine 'the extent to which young children are consciously aware of their own thinking processes' (p. 69). A key finding of the study was that young children are able to verbally articulate the processes involved in their own learning: 'there is evidence here of young children being consciously aware of strategies to employ and when to employ them' (*ibid.*, p. 76). The creative thinking described in Tanner et al.'s study occurs in the Foundation Phase. Some educational critics (*see* Robinson, 2009) have suggested that this kind of thinking is lost in later phases in the education cycle, pointing to a perceived narrowing of education with an increased focus on standardized testing. Nevertheless, the finding of metacognitive awareness observed in the Foundation Phase serves to support Papert's assertion that learners are indeed able to reflect on their own thinking. Specific attention now needs to be directed towards Papert's claim that Turtle Graphics can build metacognitive awareness with particular efficacy.

A study by Simmons et al. (1993) has examined the impact of LOGO programming activities on problem solving processes. Papert makes the claim that Turtle Graphics is able to build a portfolio of high-level problem solving strategies ranging from debugging to modularization. Simmons et al.'s paper, however, has shown that the visual on-screen feedback provided by LOGO is more effective at encouraging low-level problem solving skills such as trial-and-error. Turtle Graphics, then, is not as effective when it comes to building high-level skills in solving problems such as abstraction.

The Simmons et al. study revealed little evidence, in other words, that Papert's Turtle Graphics leads to the gains that were suggested in *Microworlds*:

> Trial-and-error attempts to a solution are supported […] it appears to be easier to use low-level procedural knowledge to generate action towards a solution than to think out a hypothesis which also requires the use of conceptual knowledge (*ibid.*, p. 175).

The findings by Simmons et al. (1993) focusing on problem solving are further supported by Ennis (2012). Though not specifically addressing the claims by Papert, Ennis observes that the act of programming a computer does not in itself build thinking skills.

According to Ennis, programming activities are only able to *facilitate* the development of these skills when the programming environment is supplemented with mediation by an education professional. The square-triangle-house paradigm exercise, for example, may not flow in the way Papert describes without careful teacher invention: many learners may try to change the procedures 'triangle' and 'square' rather than creating a new procedure 'house' comprised of both. The teacher, in other words, needs to *mediate* the programming activities.

> No programming language intrinsically teaches or enhances a student's problem solving ability […] problem solving skills are not automatically or easily transferred to another domain […] Programming instruction *can* [italics in original] facilitate problem solving or thinking abilities, but this does *not* [italics in original] occur without specific problem solving instruction, direct teacher guidance, mediated practice (*ibid.*, p. 20).

Papert's expertise and excitement with Turtle Graphics could have led to an improvement in problem solving skills. But research suggests that in order for this success to be replicated in other classrooms, a similar level of attention needs to be given to the design of an effective learning environment and its pedagogical implementation. Laurillard adds further support to this idea by articulating the importance of pedagogic design clearly: 'new technologies invariably excite a creative explosion of new ideas for ways of doing teaching and learning, although the technologies themselves are rarely designed with teaching and learning in mind' (2009, p. 5).

A final point worth making is that much of the research reviewed here so far relates to US educational thinking. LOGO and microworlds were primarily US developments and as such were principally aligned in research studies to US educational thinking. For a British perspective, see Howe et al. (1982a) where a project in the early 1980s saw a microworld developed for the domain of mathematics. We will also consider work by Sharples (1985) in some detail later in this review that also provides a British perspective.

To summarize, Papert makes the bold claim that ideas such as debugging and modularization are developed when programming with Turtle Graphics. We have considered some of the research that questions this claim in this section. Similar criticisms are considered when discussing computational thinking later in this review.

Ennis (1992) suggests that it is careful mediation by the teacher when a learner is programming that has the potential to build thinking skills. Despite Papert's excited observations of the Brazilian samba schools, even Papert (1980a) seems to align himself with this point by suggesting that the teacher should make carefully timed interventions throughout the learning process. Ennis (1992) clarifies this point by explaining that it is the role of the teacher to emphasize the thinking skills that a learner needs to know as they (learners) carry out programming activities. He suggests that no programming language improves thinking skills in itself: direct teacher instruction is required about the nature of the skills that are being learned. Programming education in other words, should exist as a 'mediated practice' (p. 20) where learner-centered activities are combined with direct teacher input.

2.3 Moving Further with Microworlds: Resnick, StarLogo and Rieber

The microworld, as Papert (1980e) puts it in the title of one his articles, is an *incubator* for powerful ideas because it allows ideas to grow in a *natural*, Piagetian way. Using Papert's incubator analogy, the microworld of Turtle Graphics aimed to incubate ideas in the domain of mathematics. This section explores how the microworlds-based approach to learning has developed since the invention of the initial Turtle Graphics microworld.

2.3.1 Decentralized Systems and Self-Organising Behaviours

In *Turtles, Termites, and Traffic Jams*, Resnick (1997) sets out his plan to develop a microworld that is designed to foster a different kind of idea. For Resnick, microworlds provided an opportunity to investigate *decentralized systems* and the self-organising behaviours that emerge within them. Whereas the domain for Papert's microworld was Euclidean geometry, the purpose of Resnick's microworld was to investigate decentralized systems. Before this can be discussed further, a detour is required in order to establish what *decentralized systems* means.

According to Resnick, a discourse of centralization has traditionally prevailed within our culture through what he terms 'the centralized mindset' (*ibid.*, p. 4). The centralized mindset assumes a central authority, one single cause, behind each pattern of behaviour which is perceived in Western culture. Everything which happens, put simply, is assumed to have some kind of unified, ultimate explanation. A central theme of Morozov's (2013) *To Save Everything, Click Here* is something he terms *Internet-centrism*: 'once part of "the Internet" technology loses its history and intellectual autonomy [… the Internet] appears fixed and permanent, perhaps even ontological' (p. 18). According to Morozov, there is a tendency to focus on a stable, central Internet (to a postmodernist, a metanarrative) to the detriment of the competing technologies, web sites and ethical concerns that comprise it (the mini-narratives). Resnick (1997) explains this idea further:

> [P]eople seem to have strong attachments to centralized ways of thinking. When people see patterns in the world (like a flock of birds), they often assume that there is some type of centralized control (a leader of the flock). According to this way of thinking, a pattern can exist only if someone or something) creates and orchestrates the pattern (p. 4).

If the centralized mindset assumes a central authority behind patterns or behaviour, how would one describe a *de*-centralized mindset? What does decentralization have to say about how Resnick's flock of birds manages to navigate itself through the sky with such precision? There is no leader of the flock or controlling authority as posited by those subscribing to the centralized mindset. Instead, the appearance of a unified, structured whole is an *emergent* behaviour from a series of smaller, simpler interactions. Emergent properties, according to Christakis and Fowler (2011), occur where attributes of a whole emerge from the interconnection of parts (p. 26). There is no bird in the flock which controls the action: no single bird orchestrates the movements of the rest. Every bird simply reacts to what its neighbours are doing. When all these smaller interactions are put together, a self-organized ordered flock *emerges.*

Braitenberg (2001) carried out a series of thought experiments to demonstrate the power of the centralized mindset. Braitenberg's vehicles were small robots fitted with a basic set of sensors as inputs and motor outputs to enable movement. Braitenberg then programmed the vehicles with a set of basic rules such as 'slow down when a lot of light is detected' and 'slow down when not much light is detected'. What Braitenberg noticed was that the disparate, simple interactions led to complex behaviours in a similar fashion to Resnick's flock of birds. He suggests that the emergent complex behaviours would lead the viewer to postulate some kind of central structure such as 'the creature is scared of the dark' and ignore the possibility of smaller, local interactions.

Resnick set out to design a microworld called StarLogo. The unique thing about StarLogo was that it provided a microworld within which learners may explore the workings of decentralized systems and the emergent behaviours that they give rise to. The construction of the StarLogo microworld differed from that of the Turtle Graphics microworld in several important ways. It is to these differences that the next section turns.

2.3.2 StarLogo: A Microworld for Exploring Complexity-from-Simplicity Situations

One of the key features of the Turtle Graphics microworld was that there was a single graphical Turtle. The Turtle was able to move forward, back, left and right; lift a pen up and down off the ground to control whether his movements leave a mark; even commit extremely complex sets of instructions to memory.

For Resnick, the thing that made Turtle Graphics so effective, however, simultaneously presented the biggest barrier for finding out about decentralized systems. The Turtle Graphics microworld was simple: it involved a single Turtle that moved about the screen. But as we seen in the previous section, decentralization is about the emergence of an organized whole emerging from lots of smaller interactions. Resnick provided a solution to this problem he faced:

> StarLogo has *thousands* [*italics in original*] of turtles. And StarLogo is designed as a *massively parallel* language – so all of the turtles can perform their actions at the same time, in parallel (*ibid.,* p. 33).

The first significant departure from the microworld of Turtle Graphics, then, is that there is more than one turtle running in parallel. Just as parallel processing splits up its operations into different tasks that may be carried out *at the same time*, StarLogo is a parallel language. Each Turtle can represent a single bird in a flock, and each of the Turtles can be programmed with the same behaviours to carry out simultaneously.

Second, the very nature of the StarLogo Turtle is different to the Turtle Graphics Turtle. Whereas the Logo Turtle was designed to carry out geometrical drawings, the StarLogo Turtle was designed to carry out behaviours. Each StarLogo Turtle is able to detect nearby Turtles and follow a scent ingrained into 'patches' in the StarLogo world. Each StarLogo Turtle is able to interact with one another and interact with the environment in which it inhabits:

> [Turtles] can detect (and distinguish) other turtles nearby, and they can "sniff" scents in the world. (*ibid.*)

The fact that the Turtle can interact with certain conditions in its environment illustrates a third difference between the Turtle Graphics microworld and the StarLogo microworld. With Turtle Graphics, the environment takes a somewhat passive role. The environment is merely a blank canvas of pixels onto which the Turtle may use a pen to create a geometric drawing. The StarLogo environment, on the other hand, is comprised of a series of *patches.* Patches are just like Turtles in that they can record information and initiate commands: the only difference is that patches cannot move around the screen whereas Turtles can. In StarLogo, each individual Turtle is responsive to the behaviours of all of the Turtles around and is constrained by those behaviours. StarLogo, in this way, brings the environment to life. As Resnick explains:

> By reifying the environment StarLogo aims to change the way people think about creature-environment interactions […] Instead of communicating with one another directly, StarLogo creatures can communicate indirectly through the environment, by releasing chemical pheromones into the patches. […] All types of interaction are possible: turtle-turtle interactions, turtle-patch interactions, patch-patch interactions (*ibid.*, p. 34).

A table comparing the different properties of computational objects in the Turtle Graphics and StarLogo microworlds is shown in *Table 1*.

Microworld	**Computational Object Properties**		
	Number of Computational Objects	**Interactions with other Computational Objects**	**Interactions with the Environment**
Turtle Graphics	A single turtle with position, heading and pen state. Turtle as a *mathematical* object.	Support for one turtle only.	Environment records drawings but does not interact with turtle.
StarLogo	Supports many turtles carrying out behaviours *in parallel*. Turtle as a *dynamic* 'swarm' type object.	Turtles can interact with turtles in the nearby locality and detect the scent they leave.	Screen divided into *patches* that can interact with turtles and initiate commands.

Table 1: Computational Objects in Turtle Graphics and StarLogo

The title of Resnick's book *Turtles, Termites, and Traffic Jams* refers to three activities that may be carried out in StarLogo to explore the phenomena of turtle population levels, termite nest construction and traffic flow around traffic enforcement cameras. The first scenario investigates the impact of differing populations and food levels upon an ecosystem of turtles. The second investigates the formation of piles of woodchips when increasing or decreasing the number of termites. The third examined the behaviour of traffic jams, both in the presence and absence of traffic enforcement cameras.

In all three situations, it was possible to see the emergence of some kind of pattern as a result of a few simple low-level interactions carried out in parallel. Resnick created StarLogo to examine how 'complex, orderly patterns [may arise] from interactions among simple, homogenous objects'. It is this very notion that is integral to decentralized systems and it what Resnick terms a 'complexity-from-simplicity situation' (*ibid.*, p. 46).

In summary, the StarLogo microworld differs from the Turtle Graphics microworld because it enables multiple Turtles to interact with their environment and each other. In so doing, it is possible to explore how several simple interactions can combine together to form the appearance of an ordered, patterned whole.

Thus far in this chapter, we have examined two seminal microworlds in detail. The first microworld, Turtle Graphics, was designed to facilitate the acquisition of ideas in the field of geometry. The second microworld, StarLogo, was designed with a different idea in mind: it was created to enable learners to find out about decentralized systems and the self-organizing behaviours that emerge as a result.

The remainder of this chapter aims to bring together the two microworlds considered so far by looking at an attempt that has been made to form a unified conceptual framework. It is hoped that this framework will clarify what a contemporary microworld looks like in practice.

2.3.3 Rieber's Framework for Microworlds

> Educational computing […] has much to gain by the infusion of constructivism into instructional design […] (Rieber, 1992, p. 94).

As was discussed in depth earlier in this review, Instructionism is the theory which says that to improve learning, teaching needs to be improved. It is the teacher who determines the curriculum and prepares the path of learning. Piagetian Constructivism, on the other hand, supposes that learners construct their own forms of knowledge through the mental processes of assimilation and accommodation. Learning happens, put simply, when learners are exploring the environment around them. Papert's Constructionism takes this idea a step further by suggesting that the construction of mental models happens particularly effectively when learners are designing, constructing and reconstructing products in the real world.

Rieber's opening quotation to this section models the microworld as a *bridge* between two theories of learning: an *infusion* of two paradigms. The microworld represents a compromise between Instructionist and Constructivist/Constructionist approaches. Instructionism calls for a curriculum of learning that is closely marked out by the teacher. Constructivism and Constructionism require a more open-ended approach to learning that is characterized by a model of exploration. The microworld acts as a bridge between both paradigms by proposing a model of *guided discovery.* Discovery and exploration *can* still happen, but it can happen within a constrained domain for learning. The Turtle Graphics microworld was constrained in such a way that knowledge developed in the area of geometry. The StarLogo microworld placed its boundaries differently, instead encouraging learning in the domain of decentralized systems.

More recently, Rieber (2004) has broken down this definition further and has established a conceptual framework for understanding microworlds by framing them in terms of five functional attributes (p. 588). Each of these attributes will now be discussed in turn.

2.3.3.i Attribute 1: Domain Specificity

The first attribute identified by Rieber is that microworlds are domain-specific. The seminal microworld of Turtle Graphics was designed so that learners discover powerful ideas in the domain of mathematics. In particular, Turtle Graphics was designed to cultivate an understanding in the area of geometry (Papert, 1972). By entering the directional command RIGHT 90, the learner will find that an on-screen turtle rotates around its centre through an angle of 90 degrees. One of the very most basic directional commands in Turtle Graphics, then, enables learners to explore the impact of increasing and decreasing the angle of turn.

2.3.3.ii Attribute 2: Understandable

Rieber's second attribute of a microworld is that it must be **understandable** to the learner. The microworld must have a low floor to facilitate access for learners at various stages of development. One of the key problems with the Turtle Graphics microworld is that learners are required to remember syntax. As Resnick et al. (2009) explain: 'early programming languages were too difficult to use, and many children simply couldn't master the syntax of programming' (p. 63). It is important to remember, however, that the limitations of early programming environments were subject to the technological affordances of the time period. As learning technology has become more powerful, it is now possible to design new programming tools that attempt to minimize the occurrence of syntactic errors as a barrier to learning programming.

This has given rise to programming tools like Alice: a 3D animation environment that uses visual formatting for generating code to remove the syntax problem (Cooper et al., 2003; 2004). The Lifelong Kindergarten group at MIT have tried to lower the floor further by developing a building-block programming language called Scratch. Put simply, Scratch allows the learner to piece together chunks of a program in an experience just like slotting together Lego bricks. Learners can build their own programs without having to remember the correct sequence of commands. The Scratch programming language is discussed in detail later in this review.

2.3.3.iii Attribute 3: Instrinsically motivating

Third, the microworld must be **intrinsically motivating** for the learner. Learners must remain engaged with the microworld for long enough so that they develop through stages of increasing complexity. If Rieber's second requirement was for a low floor, then it must also motivate to reach a high ceiling with added sophistication over time. Also important to motivation and engagement is 'wide walls'. Microworlds require enough scope to enable the learner to make personally meaningful projects according to their own backgrounds and interests. As Resnick (2016) explains:

> It's not enough to provide a single path from low floor to high ceiling; we need to provide wide walls so that kids can explore multiple pathways from floor to ceiling.

2.3.3.iv Attribute 4: Play

The third idea is similar to Rieber's fourth characteristic of a microworld as an agent for immersive and **playful activity**. One of the key design criteria for the Lifelong Kindergarten Group when creating Scratch was that it supported 'many different types of projects (stories, games, animations, simulations), so people with widely varying interests are all able to work on projects they care about' (*ibid.*, p. 64). Rieber thought that if projects have personal appeal, learners are more likely to immerse themselves within it. They will view the project positively as play rather than negatively as work. This immersion will enable learners to gradually increase the complexity in their creations and facilitate learning within a particular domain of knowledge.

2.3.3.v Attribute 5: Constructivist

Rieber's final attribute of a microworld is that it should be designed and implemented with a **constructivist paradigm** of learning in mind. As Rieber (2004) reminds us, this final attribute also carries a set of pedagogical assumptions external to the microworld itself (p. 588). The teacher's role, according to Rieber, is twofold. First, the teacher must balance providing challenge with providing an appropriate level of support. Second, it is the teacher's role to model the process of learning within the microworld. The focus of the teacher, put simply, is to carefully make interventions in the learning process whilst designing a microworld for effective learning. The empirical research in this thesis explores this very issue of effective microworld design. This concern is something that also resonates in Papert's (1980a) vision for constructionist learning with microworlds (p. 179).

In summary, we have looked at how the notion of a microworld has developed following Papert's microworlds of Turtle Graphics. StarLogo enabled a greater number of Turtles and an active environment to exhibit behaviours at the same time, in parallel. The learning aim was the exploration of decentralized systems and the discovery the self-organizing behaviours that emerge within them.

Using Rieber, we have been able to establish a framework for reviving the microworld in the contemporary classroom as a bridge to be built between the opposing paradigms of Instructionism and Constructivism/Constructionism. The microworld enables learners to acquire new knowledge through the act of exploration. To ensure that this learning transfers, there is also a role for a teacher here in here in bridging exploratory experiences within the microworld with the real world. An example in WP5, for instance, was the verbal explanation provided by the teacher in order to facilitate transfer from the arrangement of blocks in the microworld to poetic structures in the real world.

A microworld is carefully constrained: it should be specific to a particular domain of knowledge. It needs a low floor (to enable easy access at first) and a high ceiling (to enable progression to increasing complexity later). Above all, the microworld should enable the learner to make something that is personally meaningful to them (within the constraints of the microworld).

2.4 Coding as a Creative Enterprise

> But children, what can they make with mathematics? Not much. They sit in class and they write numbers on pieces of paper. That's not making anything very exciting. So we've tried to find ways that children can use mathematics to *make* [italics in original] something -- something interesting (Papert, ca. 1980).
>
> [A Microworld] leads to immersive activity best characterized by words such as play, inquiry and *invention* [emphasis added] (Rieber, 2004, p. 588).

Here, Papert explains his project to embed a Constructionist approach to learning within mathematics classes. He uses the idea of making. The second quotation is taken from Rieber's conceptual framework of a microworlds approach to learning and his fourth attribute for *play*. The first quotation uses the word 'make' whilst the second uses the word 'invention'. What both quotations share is the idea of some kind of generativity, design or creativity. It is clear that these ideas somehow figure into the discussions about the microworlds approach to learning.

2.4.1 Bricklaying to Playing the Cello: An Intimate Connection Between Hand and Head

Greg Gargarian used the titular phrase 'the art of design' in his PhD thesis (1996). Part of Gargarian's thesis questioned the notion of design. He wanted find out what design means theoretically. But what more can be said about design specifically in relation to programming?

In *The Craftsman*, Sennett (2009) weaves in and out of historical, philosophical and sociological modes of address to give a multi-disciplinary account of all things 'craft'. For Sennett, craft is a dialogue – an 'intimate connection' – between the hand and the head. Craft is not located in either the mind or the external world, instead it may be found in the relationship between the two.

> [Craft] focuses on the intimate connection between hand and head. Every good craftsman conducts a dialogue between concrete practices and thinking [...] the relation between hand and head appears in domains seemingly as different as bricklaying, cooking, designing a playground, or playing the cello (*ibid.,* p. 9).

It is worth sidetracking briefly to revisit an issue that was discussed earlier in this review. In section 2.1.5, we problematized formulations of constructionism as a 'learning-by-*making*' theory of knowing. It was suggested that a 'learning-by-*designing*' theory of knowing is more suited to the spirit of constructionism since the term *design* also suggests an act of intentional thinking on the part of the learner. Sennett's construction of *craft* embodies a dual sense of thinking and physical making in the same way that our operational definition of design shares both of these things. For this reason, a key idea underpinning this thesis is that craft and design may be used interchangeably.

Sennett develops the relationship between the mind and external physicality further: he suggests that 'all skills, even the most abstract, begin as bodily practices' (*ibid.*, p. 10). This suggestion can, albeit fairly loosely, be mapped onto a broadly Piagetian model of intellectual development. This is because, for Piaget, concrete thinking develops *prior to* abstract thinking. As Boden (1994) points out:

> The significance of the stages for Piaget is that their order is *invariant* [my italics], because each one is an essential epistemological preparation for those that will follow (p. 20).

Where Sennett departs from the Piagetian model, however, is that for Piaget the order of the stages is sequential. Put simply, for Piaget, progression through the stages of intellectual development is a 'one-way' process. Sennett (2009) shares Piaget's suggestion that the concrete is preparation for the abstract, but he moves on to suggest that this also works *vice versa:* 'technical understanding develops through the powers of imagination' (p. 10). For Piaget, intellectual development follows a path of *sequential* stages. For Sennett, skill development is much more *dialectical* in nature: design involves a continuous interplay between the head and the hand. This departure from the linearity of Piaget's stages is an idea shared by Turkle and Papert (1991):

> Piaget sees a progression from egocentric beginnings to a final, "formal stage" when propositional logic and the hypothetico-deductive method "liberate" intelligence from the

> need for concrete situates to mediate thinking […] In this vision, mature thinking is abstract thinking. We disagree: for us, *formal reasoning is not a stage but a style* [*italics in original*] (p. 175).

Turkle and Papert's criticism may appear harshly directed here. One could argue that Piaget makes no *absolutist* claims for the order of the stages, only their sequential epistemological layering in the majority of cases. In addition, Turkle and Papert's concept of a learning 'style' can be easily contested, not least because of its vague and abstract nature.

Sennett's work contributes to the development of this thesis because of the significance he attaches to *thinking* within the design process. Sennett's construction of design as a connection between the hand and head is something that is shared with Frayling in *On Craftsmanship* (2011). Frayling develops the idea further by proposing that design also embodies a third characteristic. His formulation of design is a tripartite structure of hand, head *and heart*. Frayling does not use this additional term to indicate emotion, but instead refers to the craftsman's understanding of cultural codes of design:

> *The heart:* [italics in original] which has nothing to do with sentiment, but rather with having one's finger on the pulse of what is going on in the broader visual culture and in the creative industries (*ibid.,* p. 143).

Frayling's work is also important for another reason. He sets up a dichotomy between two senses of learning with art. Teaching *to* art involves the learner developing an understanding of the skills necessary to become an effective designer. Teaching *through* art is where the learner develops a series of related skills such as problem solving and flexible thinking. Frayling suggested that the most effective learning experiences are when teaching *to* and *through* art happens simultaneously:

> [T]he very best form of art education should combine […] teaching *to* art and teaching *through* [italics in original] art but […] teachers ought to be clear about which they were teaching at any given time (*ibid.*, p. 140).

The parallels that can be drawn between this idea and a microworlds approach to learning are clear. Papert's Turtle Graphics microworld enabled the student to learn *to* program whilst also learning geometry *through* programming. Resnick's StarLogo microworld enabled the student to learn *to* program whilst also learning about decentralized systems *through* programming. Frayling also stresses a requirement for the teacher to establish clear boundaries about what subject content should be learned *through* the process of making an artwork. The importance of domain specificity in the construction of a microworld was discussed in section 2.3.3.i.

Nearing the end of the work, Sennett suggests how one may construct an environment for encouraging skill development. In so doing, he draws on the work of architect Aldo van Eyck.

Post World War II, Eyck started a project of transforming the empty spaces in Amsterdam with playground and park areas. The aim of Eyck's architecture was to encourage children to develop skills in safe navigation of the urban space. As Sennett (2009) explains:

> Eyck favored simple forms of play furniture that give few directions for use […] Kids have had to come up with their own game rules that permit play without being hit by cars. The architect, then, designed a park using the simplest, clearest elements that invite its young users to develop the skill of anticipating danger and managing it […] The design provides subtle guidance (p. 233).

Such a notion of skill development is aligned with a microworlds approach to learning. The architect in Sennett's example has not created a rigidly-structured play area with dedicated zones for different tasks. Nor has he chosen to transform the area with one open-plan sand pit. The architect has given a lot of thought to how he may design an environment for navigation in an exploratory way, whilst simultaneously encouraging the development of specific skills. There are some subtle constraints built into the design of the play area such as benches placed in key areas. In this example, the set of constraints are designed as such so that children may develop the skill of anticipating danger.

The role of the teacher according to such a model should be akin to the role of Eyck when designing his playground. Both aim to construct an environment which sets up the right conditions for learning through exploration. A balance needs to be struck between open-ended exploration and constraints to encourage learning within a particular area. Teaching, as Laurillard (2012) writes in the title of her work, is a 'design science'.

The similarities between Eyck's playground and the microworlds approach to learning are so similar, in fact, that it would be a fair assessment to call Eyck's playground a microworld in itself. It is interesting to note that the father of term *microworld*, Papert, also recognizes that there is no requirement for a microworld to be computer-based. Computers, for Papert (1980e), are merely a medium where particularly effective and engaging microworlds can be created:

> The [microworld] project is not new in kind. One can recognise much of the praxis of educators as falling under this description […] but *what is new is the availability of technology for microworlds* [italics in original] – a technology that will […] allow it to become a refined, systematic and theory-based branch of education research (p. 204).

2.4.2 Making is Connecting

Gauntlett (2011) uses a wide theoretical base to investigate the notion of design. In *Making is Connecting*, Gauntlett's primary argument is that Western culture is moving away from a 'sit-back-and-be-told' model, where its subjects are positioned as passive media consumers; to a 'making-and-doing' model, where its subjects are positioned as active media producers. Put simply, media content is produced by a variety of different people. Gauntlett does list some exceptions to this: one such exception is perhaps the rise of devices like the *Apple iPad* and its *App Store* – a pool of centrally controlled and vetted content. Indeed, an opposing view to Gauntlett is that we are in fact moving towards a culture that is increasingly polarized with a relatively smaller number of media producers in comparison to a large pool of consumers. Gauntlett argues, however, that we are moving in a different direction: that of an increasingly democratized – and to use Resnick's term *decentralized* – culture.

Gauntlett contextualizes this shift from passive consumer to active producer within all kinds of discourses and institutions. It is his thoughts about the education system, however, that are of particular importance here. Like Papert, Gauntlett recognizes that the dominant discourse in education has traditionally seen the student occupy a subject position of passive recipient of knowledge. This is in contrast to the teacher who occupies the active position of transmitter of knowledge. He appears to describe, in the quotation below, what Papert would term *Instructionism*:

> School education has tended to settle around a model where a body of knowledge is input into students, who are tested on their grasp of it at a later point (*ibid.*, pp. 8 – 9).

Gauntlett contrasts this vision of a bleak, Instructionist past with a vision of an improved education of the future. Again, one is able to draw parallels between Gauntlett's vision for the future and the learning ethos that a microworlds approach promotes. Such comparisons may be drawn in several ways.

> [S]tudents work on learning projects, in which their teachers encourage them to ask questions and seek out understanding for themselves. To present their learning to others, they produce exhibitions, physical performances, online presentations, and games (*ibid.,* p. 237).

First, Gauntlett's emphasis on a mode of learning characterized echoes that of many key thinkers in constructionist research. Kafai (1996b), for example, implemented a video game design project with nine-to-ten year olds over an extended timespan of six months. Crucially, students were provided with a very open brief - to create a video game to learn about fractions – and were able to take this theme in their own, personal directions. The project resulted in a variety of games being produced in genres including adventure, sport and simulation. The emphasis that Gauntlett attaches to personal projects also resonates with problem-based learning. This is the idea that emphasizes the role of practical experience in the learning process and cycles of solving authentic problems followed by periods of reflection. As Hmelo-Silver (2004) explains:

> Problem-based approaches to learning have a long history of advocating experience-based education. Psychological research and theory suggests that by having students

> learn through the experience of solving problems, they can learn both content and thinking strategies. Problem-based learning (PBL) is an instructional method in which students learn through facilitated problem solving (p. 235).

Second, the idea that these extended projects should culminate in some kind of showcase of designs has also been used by practitioners in constructionist research. A recent article by members of the Lifelong Kindergarten group at MIT (Resnick et al., 2008a) has reported on a pilot scheme for broadening participation with robotics technologies. This research led to the recommendation that an exhibition format is an effective way for children to share their design creations. An example of such an online platform for exhibiting work is the Scratch website at *scratch.mit.edu*. This website also enables learners to gain feedback from their peers via online community tools.

Third, Gauntlett (2011) explains his vision of teachers in the future: '[teachers] are no longer just the "holders of the answer book" but are visibly also learning new skills and knowledge in their own lives' (p. 237). This vision is consistent with Papert's famous example of the Brazilian samba schools, though perhaps less idealized in its scope. An important part of the samba school model that Papert (1980a) discusses in *Mindstorms* is that 'the experts are also learning' (p. 179). More recently, Kafai et al. (2008) have suggested that the deconstruction of the traditional 'teacher knows everything' power relationship is a particularly promising step for education. As they explain: 'mentoring can comprise of different interactions ranging from teaching to *learning* [*my italics*]. Mentoring should be seen as a continuum and not a set of prescribed roles' (p. 202). These ideas are not dissimilar to the Vygotskian learning approach that was discussed previously.

Fourth, with the updated role of the teacher comes a different understanding of how learning happens. Gauntlett (2011) explains that learners will less and less be 'required to regurgitate "correct" answers which have been previously dispensed by the teacher, but are expected to research and think about a subject and then produce their own responses to it' (p. 238). That all learners will approach and respond to a topic differently is something that particularly resonates in Turkle and Papert's (1991) *Epistemological Pluralism* where they emphasize that there is no 'right way' to learn programming.

Fifth, and finally, Gauntlett (2011) emphasizes that learning should be 'connected to real-world questions and problems. In particular, a substantial part of the school experience […] embedded in the surrounding environment and genuine local issues' (p. 238). Such a vision is very similar to that found in Papert's (1980b) discussion on the topic of secondary school science: 'I believe that the child's intellectual growth must be rooted in his experience' (p. 164). As we have already seen, for Papert, the syntonicity of the Turtle provides an effective means by which a learner may make this connection to their real-world experiences.

Gauntlett's points are open to challenge and should be approached critically. Some may argue that the binary opposition that Gauntlett draws between teaching and learning is overstated. Indeed, the emphasis on continuing professional development (CPD) in the British teaching profession may serve as evidence to the contrary: teaching has always involved an element of learning. It may also be argued that whilst pedagogical development is important in many areas, some elements of Instructionist pedagogy must be retained for certain curriculum areas or topics. Indeed, we have already looked at how a microworlds-based approach to learning may work to provide a bridge between Instructionist and Constructionist models (Rieber, 1992).

Finally, expounding upon the work of the psychologist Csikzentmihalyi, Gauntlett makes the following comments on the notion of *flow* when learning using a computer:

> 'Flow' describes the experience where a person is wholly engaged in a task, to the extent that time passes unnoticed, and they forget about demands external to the task […] The 'flow' experience is presented as being desirable for, and attainable by, anybody, as they go about their work (*ibid.*, p. 75).

Flow refers to a psychological state of complete immersion when undertaking a task. To borrow a colloquialism, flow is when the learner is 'in the zone': it is a state of intensified focus that is achieved when engaging with a task. *Flow* is not something that is directly mentioned by Papert in *Mindstorms*, but he (1980a) does make reference to something he terms 'holding power': 'the computer's "holding power," so feared by critics, becomes a useful education tool' (p. 27). Csikzentmihalyi (2004) explains that 'flow' can be a desirable state for learners to achieve because it describes a period of intrinsic motivation where learners operate at their 'fullest capacity':

> The defining feature of flow is intense experiential involvement in moment-to-moment activity. Attention is fully invested in the task at hand, and the person functions at his or her fullest capacity (p. 230).

The immersive state of flow negates the role of reflection within the learning process. This tension is important point because, as Laurillard (2002) suggests, reflective activity is an integral part of teaching and learning: 'academic teaching must help students *reflect* [*my italics]* on their experience' (p. 24). It may also be necessary for learners to hold a base level of skill within a domain of knowledge in order for the psychological state of flow to be achieved when carrying out an activity with greater independence. The implications of flow upon pedagogical practice are not considered as part of this project but an interesting area for future research could focus on the relationship between flow and microworld-based learning approaches.

Thus far in this section, we have looked at the notion of design from a very broad perspective. Parallels have then been drawn between the act of designing and the microworlds-based learning approach that underpins this thesis. The next section of the chapter moves from this broad perspective to home in on one or two design activities in particular: the art of computer programming.

2.4.3 Creative Programming Projects

> The Apple II, as the immediate product of guys who love to hack and tinker with technology, was a machine which, like many early computers, didn't do anything when you switched it on, but waited for you to do something with it (Gauntlett, 2011, p. 177).

In *Making is Connecting*, Gauntlett draws on Jonathan Zittrain's notion of *generative technology*. When early desktop computers like the Apple II first started making their way into the home, interacting with technology was an experience defined by *generativity*. Generativity, as Gauntlett puts it, is exercising 'the right and opportunity to be creative' (*ibid.*, p. 176). Early personal computers required a great deal of imagination and technical skill if they were to be put to good use. This is because they did not arrive with software 'pre-programmed' by experts: the hardware was not designed to work straight out of the box with minimal user effort. Early computers like the Apple II were built with the assumption that the *user* would program it to carry out the tasks that were desired.

This is an idea, of course, which contrasts some recent technological developments. The iPhone, iPad and Apple Macintosh platforms all contain 'app stores' where users can download ready-made apps from expert developers to their devices. Crucially, Apple controls which apps it will allow and which apps it will not allow on its distribution network. When he first announced the 'App Store', the late Steve Jobs (Apple Inc., 2008) commented light-heartedly: 'will there be limitations? Course. There are gonna be some apps which we're not going to distribute [*sic*]'. This is a trend which has continued with Microsoft's built-in 'Windows Store' as part of their latest operating system releases.

The activity of programming a computer, put simply, repositions the user in relation to defining the actions of the computer. Programming is a generative activity because it enables the user to take control of their own technological designs ranging from interactive stories to video games. As Gauntlett (2011) explains, computer programming offers 'a huge open door to creativity and innovation' and is 'not a "back-stage" activity' (p. 178). Programming empowers the user to tinker with the computer in order to pursue their own creative projects.

Gauntlett is not alone in this view that programming is fundamentally a creative activity. Kafai and Peppler (2005) have argued that programming a computer is a means for personal expression; an assertion that builds on the work of John Maeda in his book *Creative Code* (2004). Computer programming has traditionally been viewed as an activity associated with mathematics and science classrooms. What Kafai and Peppler suggest, however, is that programming can also be used for artistic and creative projects:

> [P]rogramming projects tend to focus on mathematical and science content. As an alternative, we emphasize programming projects that use graphic, music and video […] making them more amenable to an arts environment (*ibid*, p. 1).

By undertaking creative programming projects ranging from animation to video game design, Kafai and Peppler argue that two types of learning takes place. First, learners are able to build an understanding of key computer science concepts such as program assembly, looping and conditions. Second, they learn to appreciate computer-based art forms and how to express themselves using media arts. Learners are able to gain an understanding of how to program a computer *and* how to create media arts. In so doing, they are simultaneously reflecting on existing media arts and therefore learning how to *appreciate* them. As Kafai and Peppler explain:

> [I]t is not just in the viewing or playing but also in the constructive – or coding – experience that connections to art can be established. In other words, coding allows the creator to simultaneously *appreciate* and *create* [my italics] art (*ibid.*, p. 5).

The significance that Kafai and Peppler attach to understanding the *construction* of digital media products resonates in Rushkoff (2010). In *Program or be Programmed*, he argues that an understanding of how technology is programmed is necessary within an increasingly digital culture. Deeply engrained within digital media products are the 'biases and agendas' (p. 11) of the programmers and the programming houses where they work. His argument is that we need to engage critically with how digital media products are created in order to problematize, rather than naturalize, such biases and agendas. This is essential if we are to avoid being 'programmed' by the programs:

> When human beings acquired language, we learned not just how to listen but how to speak. When we gained literacy, we learned not just how to read but how to write […] In the emerging, highly programmed landscape ahead, you will either create the software or you will be the software (*ibid*., p. 7).

This is an important idea, but further empirical research is needed to explore the extent to which this is the case. If evidence supports the assertion, then such an idea is not dissimilar to the microworlds based approach and the examples we have considered so far. With LOGO Turtle Graphics and StarLogo, learners are learning to program whilst concurrently learning about concepts in mathematics and geometry (Turtle Graphics) and decentralized systems (StarLogo). Kafai and Peppler suggest that the learner simultaneously develops an understanding of the cultural codes and conventions that Rushkoff discusses here through making their own computer programs using the Scratch platform.

One case study provided by Kafai and Peppler is that of 13-year-old Kaylee. Choosing to make a dance video, Kaylee makes design decisions about which music track to use, which images are 'cool' and what animated dance moves to recreate. In so doing, according to Kafai and Peppler, Kaylee reflects upon the cultural codes and conventions of the music video as a media product. Alongside the *computational thinking* dimensions that learners encounter (such as those described in Curzon et al., 2014), a further framework of *cultural* codes is engaged with.

The notion of creative programming has also been extended to creative robotics. Resnick et al. (2008) provide a useful definition of what *robotics* means. Within culture, and perhaps other academic disciplines, there is a tendency to link robotics with notions of AI (Artificial Intelligence). The definition provided by Resnick et al., however, is much broader and all-encompassing in scope. Robotics is any activity where machines are programmed to carry out actions based on sensor inputs. According to this definition, a burglar alarm system is an example of robotics because it relies on PIR (passive infra-red) inputs to sound a siren output. A greenhouse heating system is also an example of robotics because it is reliant on heat sensor inputs to control an actuator for the opening and closing of windows. As they explain:

> Increasingly, robots are becoming available as consumer products, such as the Roomba vacuum-cleaner robot and the Aibo walking robotic dog. But these stereotypical images can be misleading. Walking and rolling robots represent just one type of robotics. *Robotics includes all types of programmable machines that perform actions based on inputs from sensors* [my italics] —everything from a home security system that sounds an alarm when it detects motion to a greenhouse that regulates its temperature and humidity (*ibid.*, p. 59).

Readers who are familiar with the UK education system would quite rightly see overlap here with what has become known as *control systems*. Indeed, the *Design and Technology in the National Curriculum for Wales* (DCELLS, 2008a) document places the requirement to 'understand feedback' and 'achieve different kinds of movements in products' under the banner of working with systems and controls (p. 15). For the purpose of this review we will use Resnick's broad formulation as an operational definition of robotics. Robotics, then, is *programming machines to carry out tasks* with *inputs from sensors*.

Part of the educational appeal of creative robotics, for Resnick et al., is that is involves design skills in two ways (*ibid.*, p. 60). First, creative robotics involves *physical* design in the real-world. If the learner decides to build a fan that automatically turns when someone comes near, then a key design consideration will be the appropriate placement of the motor and light sensors within a given physical space. Second, creative robotics involves *computational* design in the virtual world. The learner must design the coding of the behaviours on the computer so that the real-world creation reacts to the inputs as desired.

By undertaking creative robotics projects, Resnick et al. argue that learners are able to mix art and engineering in a way that has traditionally been kept segregated in schools. In order to make their fan creations work, learners need to understand the computer science concept of conditional structures to check when someone comes near. The fan must carry out this task continually so that the fan stops when no-one is near, thereby introducing the concept of looping structures. These structures must be placed within the correct order, however, thereby introducing direct sequencing. All of these computer science and engineering concepts will be accompanied by creative real-world design:

> They use a wide variety of materials (including felt, construction paper, pipe cleaners, and LEGO bricks) to create flowers, insects, and hanging lanterns. One group writes a computer program to make the wings of a bee flap up and down […] Combining craft materials, mechanical parts, and programmable devices can inspire both girls and boys to think more creatively about what is possible and what they want to create (*ibid.*, pp. 59 - 61).

In summary, we have briefly examined the notion of *design* by moving from a general perspective to one that is more focused specifically on the area of programming. Sennett's formulation of design suggests that creative coding involves a dual sense of thinking and practical making. Gauntlett considered a shift from passive consumer to active producer in his vision for a better education in the future. Programming and robotics relates to design by repositioning the user in control: it is a generative activity that has the potential to allow for creative expression to surface. It has the potential to enable the learner to combine art and creativity with the acquisition of key concepts in computer science.

Thus far, this literature review has examined the educational ideas underpinning microworlds and looked at some early examples of microworlds in action. It then focused in on one characteristic of microworlds in particular: the notion of design. We will now move to look at the recent revival of the microworlds approach and use the historical grounding covered so far as a springboard for more recent research in this area.

2.5 Block-Based Programming Approaches

In the 1960s, Papert and his team at the MIT pioneered the microworlds approach to learning with the development of the LOGO programming language and the invention of Turtle Graphics. We have already seen how such an approach was inspired by two key epistemological models: Piagetian Constructivism and Papertian Constructionism. Resnick (2002) explains how the computer sits within such a learning approach:

> Consider the following three things: computers, television, finger paint. Which of the three doesn't belong? For most people, the answer seems obvious: "finger paint" doesn't

> fit […] But until we start to think of computers more like finger paint and less like television, computers will not live up to their full potential (*ibid.,* p. 33).

The computer in a microworlds approach to learning, in other words, is the 'material' (*ibid.*) – the blank canvas – for *designing* and *making* things. And, for Resnick, it is through these processes of designing and making that the most powerful learning takes place. To use Resnick's examples: by designing with finger paints, learners acquire knowledge in colour mixing; by building with blocks, they learn about physics and structure; by making coloured bracelets, they learn about symmetry and patterning. What makes a computer such a powerful 'material', however, is its flexibility. A computer can change to accommodate all kinds of different concepts based upon the desire of the educator. Subtle changes in the 'material' of a computer-based microworld can target different content within a wide range of different domains.

The Lifelong Kindergarten Group at MIT has developed *Scratch* as a programming tool that sits well within this microworlds approach to learning. Before programming with Scratch is considered further, however, a brief detour needs to define programming more carefully.

2.5.1 Defining Programming

To begin to define programming, a distinction needs to be made between two different ways of working with a computer. The *Computing at School* (2012a, 2012b) working group provides a useful starting point when discussing the disciplines of Information Technology and Computer Science. Whereas Information Technology is concerned with the *use* and *application* of computer systems, Computer Science is concerned with the *design* and *development* of computer systems. Programming techniques, according to such a distinction, fit within the latter discipline.

The *Computing at School* (2012c) group further attempted to define what programming in educational contexts means by developing a possible Computer Science Curriculum for schools. It is important to note, however, that this remains distinct from the statutory Computer Science curriculum in England and the ICT curriculum in Wales. The curriculum document was endorsed by BCS, The Chartered Institute for ICT, as well as the key industry leaders Microsoft and Google. Programming activities should, according to this framework, incorporate opportunities for learners to experience seven key programming concepts. These concepts are outlined below:

- *Sequencing* – moving step-to-step through a program;
- *Selection* – Use of 'if-then-else' or conditional statements;
- *Repetition* – Use of loops and recursion;
- *Procedures* – Use of reusable abstractions, with parameters;
- *Inputs/Outputs* – Interaction with the program;
- *Debugging* – locating and correcting coding errors;
- *Reflection* – Self-assessing accuracy and fitness for purpose.

Robotics activities, which we consider later in this chapter (*see* section 2.5.3) also fits into this framework of programming by providing a series of *real-world* inputs or outputs that interact with the above programming concepts (Resnick et al., 2005a). Robotics, in other words, takes place where the learner uses processes of sequencing, selection, repetition and procedures to control real-world outputs or react to real-world inputs. Several case studies have highlighted the diversity of forms that this may take, ranging from a moving arm that can strike a xylophone to a model house that automatically lights in the dark (*see* Resnick, 1991).

2.5.2 Programming with Scratch

> We wanted to develop an approach to programming that would appeal to people who hadn't previously imagined themselves as programmers. We wanted to make it easy for everyone, of all ages, backgrounds and interests, to program their own interactive stories, games, animations, and simulations, and share their creations with one another (Resnick et al., 2009, p. 60).

The above quotation presents the rationale for *Scratch*, a creative programming tool developed recently by the Lifelong Kindergarten Group at MIT. This overarching aim of the Scratch project is broken down further into three main objectives that its creators argue makes it distinct from other educational programming environments. First, Scratch was designed to be more tinkerable, enabling learners to form continually-evolving goals as they progress through their projects. Second, it was designed to encourage more personally meaningful creations that allowed for a large degree of personalization and diversity of outcomes. Third, Scratch needed to be linked to an online community to enable sharing and collaboration on a large-scale (*ibid.*, pp. 63 – 65).

In a conference to Japanese educators, Resnick et al. (2004a) use a framework of five core features to further illustrate the primary design considerations of the Scratch project. A discussion of these features serves as a useful framework for explaining the BYOB activities carried out in the empirical work of the project. Each of these features will now be considered in turn.

2.5.2.i Building-block Programming

Figure 2 shows the layout of the Scratch user interface. The *blocks palette* runs down the left-hand side of the screen and spans the entire Scratch window. The blocks palette contains visual representations of pieces of code that can be slotted and placed together in the centre *scripting area*. The top-right of the screen contains the *stage*. This is where the scripting actions will be displayed. Each script that the learner creates by snapping a sequence of blocks together must be attached either to the stage or to a *sprite* (a computational object). Thumbnails of sprites in a project are contained in the *sprite list* at the bottom-right corner of the screen.

A key requirement when designing the Scratch platform was a ‘low floor’ to facilitate access for learners at various stages of development. One of the key problems with the early Turtle Graphics microworld was that learners were required to remember syntax. As Resnick et al. (2009) explain: ‘early programming languages were too difficult to use, and many children simply couldn’t master the syntax of programming’ (p. 63). The Lifelong Kindergarten group attempted to lower the floor by developing what may be described as ‘a building-block programming language’. Put simply, Scratch allows the learner to piece together chunks of a program in an experience just like slotting together Lego bricks. Learners can build their own programs without having to remember the correct sequence of commands. This works to address Rieber’s (2004) requirement for microworlds to be understandable to uninitiated programmers.

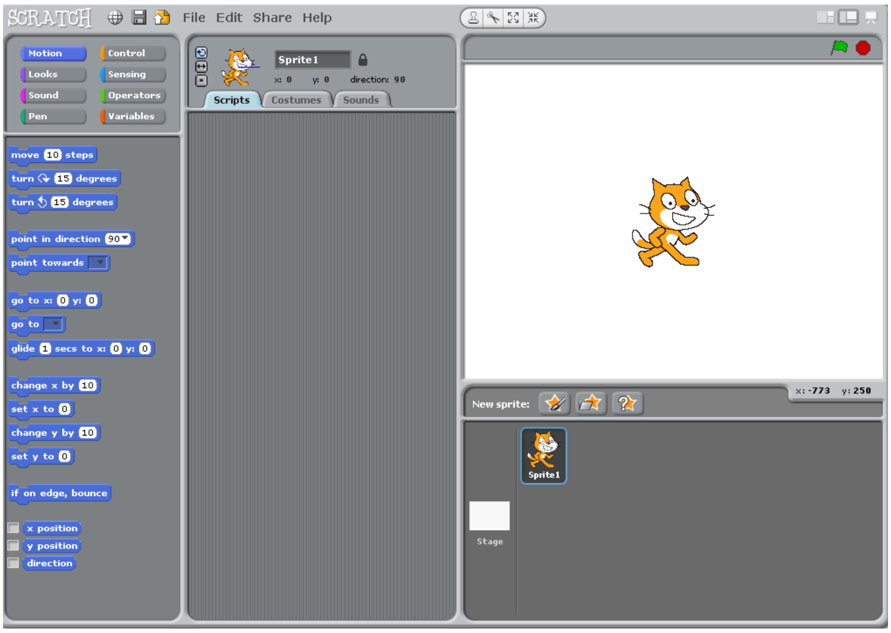
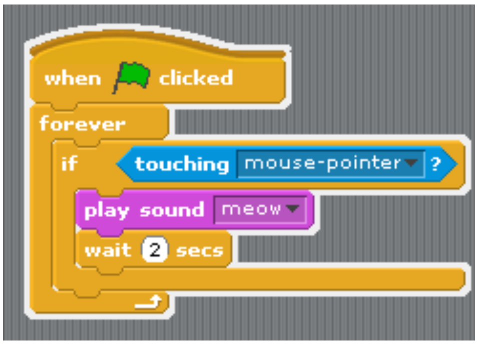


Figure 2: The Scratch User Interface

Sprites can take on many different appearances and *Figure 3* shows a Scratch script attached to a sprite that is wearing a cat costume. It shows the script required to make the cat sound a ‘meow’ and wait a few seconds each time it detects a mouse pointer over it. As with physical LEGO blocks, each virtual block of code is shaped differently. The shape of each type of block constrains how they may be snapped together, thereby greatly reducing the number of syntax errors. The control ‘when green flag clicked’ block is only ever used to start a script; it is ‘hat-shaped’ so that no other blocks can snap in front of it. The iterative ‘forever’ block is C-shaped to allow other blocks to be placed inside it. The ‘if’ conditional is also C-shaped to enable the learner to determine what will happen when the logical test returns a true value. Additionally, the logical test requires a Boolean (yes or no) input to operate and so it only allows hexagonal sensing blocks which return either a true or false value. This means that 'Scratch blocks are shaped to fit together only in ways that make syntactic sense' (Resnick et al., 2009).

A characteristic of a microworld that is shared by both Papert (1980a) and Rieber (1992) is that it must sit well within a Constructionist learning approach. The Scratch microworld provides a series of blocks that can be slotted together by learners in a number of combinations, with no restriction over design outcomes. But this platform for open-ended exploration is constrained by the different shapes and commands of the Scratch blocks. A focus on personally meaningful design projects is aligned with the principles of Constructionist pedagogy whilst the level of constraint provided by the toolkit of blocks maps to a microworld model of guided discovery.



```
when 🟩 clicked
forever
  if touching mouse-pointer ?
    play sound meow
    wait 2 secs
```

Figure 3: Arrangement of Programming Blocks in Scratch

The experience of slotting together programming blocks is integral to the Scratch microworld. This is an experience that learners can relate to their own real-world Lego style constructions. This improves accessibility – an important attribute of microworlds. It is also syntonic with prior learner experiences. It is an important caveat, however, that this syntonicity does have a degree of cultural specificity to the West: it relies on the learner having previous experiences with Lego.

The concept of the sprite is also vital to the microworld that Scratch embodies. A key part of what made Turtle Graphics a microworld, for Papert, was the syntonicity of the Turtle. The Turtle Graphics turtle can move, turn and leave a pen trail in the same way that a learner can. The Scratch sprite retains this syntonicity because it can 'step' across the screen, change a 'costume' and make a sound in an 'me-like' way that the learner can connect with.

The Scratch platform may also go even further by embodying a third mode of syntonicity. We have seen in section 2.4.3 how Kafai and Peppler (2005) link the experience of programming in Scratch to learner experiences as digital media consumers. In this way, perhaps Scratch is syntonic with learner experiences of watching animations and connects with their experience of playing video games.

Microworld	Computational Object Properties		
	Number of Computational Objects	Interactions with other Computational Objects	Interactions with the Environment
Turtle Graphics	A single *turtle* with position, heading and pen state. Turtle as a *mathematical* object.	Support for multiple turtles.	Environment records drawings as pixels but does not interact with turtle.
StarLogo	Supports many turtles carrying out the same behaviours *in parallel*. Turtle as a *dynamic* 'swarm' type object.	Turtles can interact with turtles in the nearby locality and detect the scent they leave.	Screen divided into *patches* that can interact with turtles and initiate commands.
Scratch	Several *sprites* can carry out different behaviours. Sprites can also be duplicated to carry out the same behaviour, leading to an *emulated parallelism.*	Sprites can sense other sprites and can initiate behaviours based on the states of other sprites.	The *stage* is able to initiate commands and is able to interact with the different sprites.

Table 2: Computational Objects in Turtle Graphics, StarLogo & Scratch

The model of computational objects in Scratch is compared to that of Turtle Graphics and StarLogo in *Table 2.*

2.5.2.ii Programmable Manipulation of Rich Media

Traditional notions of programming that involve working with numeric or text strings, arrays and tables have been replaced in Scratch with interactive media programming of images, sounds and animation. Scratch has been designed as part of a move to an increasingly participatory culture where the lines between media consumption and production have become blurred. We have already examined how the Internet, and in particular web 2.0 technologies, position the user radically differently to traditional media. This is what Gauntlett (2011) refers to as a shift from the 'sit-back-and-be-told' model to the 'making-and-doing' model.

Scratch enables the learner to participate in the democratization of interactive media by providing a programming platform with a rich multimedia context. Learners can create their own sprites or objects using the built-in Paint Editor with image manipulation tools such as scaling and rotation. They are able to animate the sprite by snapping together a sequence of *turn* and *move* blocks; use narrative tools like speech and thought bubbles via *say* and *think* blocks; create their own music by specifying instruments, notes and beats with *sound* blocks.

Kafai and Peppler (2007a) suggest that by using Scratch to pursue their own creative projects, learners gain an understanding of conventions in media design. Learners are able to critically reflect upon and make sense of the genres, conventions and values that are attached to media products that they consume on a daily basis. They suggest that through creating their own interactive media products, learners develop a critical eye for the codes and constructions within media culture at large. As Kafai and Pepper explain:

> [Y]outh find it hard to articulate what makes a particular videogame or software application 'good'. Asking youth to design software and/or videogames challenges them to make these assumptions explicit and asks them to build upon this knowledge to make informed suggestions for change (*ibid.*, p. 153).

In a different work, Kafai and Peppler (2007b) have explored how video game design using Scratch can integrate learning in two different contexts of computation and the arts. One case study that the paper refers to is of a learner called Jorge, a fifteen-year-old boy who participated in a video game design project using Scratch. Jorge set out to create a video game titled 'Metal Slug Hell Zone X'. Jorge's choice of video game was based on the *Metal Slug* series that he enjoyed at home in his spare time.

The qualitative study reported that Jorge was able to improve his knowledge and understanding in computation in several ways. Jorge was able to develop his skills in user interface design and learn about the importance of user-friendliness as a factor affecting technological efficiency: '[he] redesigned his program several times, discovering that it can be friendlier to the user if the game responded to standard key strokes (e.g., right and left arrow keys) rather than random ones' (*ibid.*, p. 373). Jorge developed his skills in digital graphics and animation by creating a series of still images in sequence in order to achieve a smooth stop motion animation. Jorge also gained an understanding of the world of work regarding interactive media design. He networked with others game designers, to gain specialist advice from those more experienced in certain areas, as well as with online testers to 'try out' his game (*ibid.*).

Beyond these gains in the area of technology, Kafai and Peppler argue that participation in the project also enabled Jorge to develop his artistic and creative skills. Video game making provided a context for acquiring skills in computation through the activity of programming. It does this whilst simultaneously enabling a deeper understanding of conventions and norms in interactive media design: something that Kafai and Peppler term 'games literacy'. To illustrate this point, they explain how the video game design project enabled Jorge to engage with - and problematize - notions of artistic genre:

> [Jorge] has explored and in some senses reformulated genre conventions of shooter games. The title denotes that it is in the same series as other *Metal Slug* [*italics in original*] games, yet there seems to be a parodic edge […] while he has conformed to most of the trademarks of the series, Jorge's recreation has an almost Zen-like impact on the viewer/player (*ibid.*).

It is necessary once again here to stress an element of critical distance. First, the small-scale case study approach presented in Kafai and Pepper's paper, though interesting, makes generalizability of findings particularly difficult to achieve. Second, the study was qualitative in nature with no quantitative data provided to support the gains in 'games literacy' that Kafai and Peppler suggest. Third, there is little description about the role of the teacher in the learning process and the scaffolding that was required.

The paper is useful, however, as it shares a key concern with this research project. The learning activity of programming in Scratch, Kafai and Peppler argue, is able to lead to gains beyond programming and into understanding of artistic genre. They suggest more can be learned *through* the act of programming *in addition to* how to program. This assertion is fundamental to a microworlds-based learning approach and is a principle question underpinning this thesis.

2.5.2.iii Deep Shareability

Scratch was designed with collaboration in mind and a key part of the Scratch project has been the development of a global platform for cooperation. As Resnick and Monroy-Hernández (2008c) explain, 'one of the main goals of the Scratch online community is to foster the idea of learning from each other by building on other people's ideas or projects' (p. 50).

The Scratch Online Community is a website that enables learners from around the world to share their interactive media creations as part of the perceived web 2.0 participatory culture. The website has proved popular: in the first two years of the website going live, fifty-thousand people uploaded their projects to the website (Monroy-Hernández, 2009). Central to the Scratch website is the notion of remixing: a creative act which, according to Mayer-Schönberger (2009), aims to 'actively recreate by recombining the existing work of others' (p. 89). Building on the work of

John Seely Brown, he suggests that a remixed creation finds its value through *bricolage*: 'value is derived from the (re)combination of its parts, not necessarily from the parts themselves' (p. 62). The code behind the projects that are uploaded to the Scratch website can be downloaded by other users. Other users can then add, delete or change programming blocks, sprites and sounds in order to build new creative combinations.

The computational thinking practice of *remixing* did not feature in the empirical work of this thesis, though the pair programming approach used in WP4 does link to Resnick and Monroy-Hernández's (2008c) idea of learners 'learning from each other'. There is evidence to suggest the paired approach to programming can increase engagement with the learning process (*see* Cockburn and Williams, 2000) and lead to the co-construction of technical skills (*see* Chong et al., 2005).

2.5.2.iv Support for Multiple Languages

The Scratch project has attempted to encourage multicultural and global collaboration. At the time of writing, Scratch is available in fifty different languages (Lifelong Kindergarten Group, 2012a). Changing languages is accomplished easily in Scratch, simply by clicking on the globe icon in the top-left corner of the screen. Such a focus on a multi-language and multi-cultural environment was an essential prerequisite for the Lifelong Kindergarten Group because the platform was designed to fit in with the MIT lab's network of *Computer Clubhouses.*

Computer Clubhouses are extracurricular clubs where learners are able to explore creative applications of computing in informal learning settings. The first Computer Clubhouse opened in Boston in 1993. By 2005, there were over one hundred Computer Clubhouses in twenty-one different countries (MIT Media Lab, 2012a). With such a rapid expansion into communities without English as a first language, the necessity for a multi-language environment became clear.

There has been some research coverage to indicate that Scratch fits well with this model of extracurricular learning that the Computer Clubhouse model promotes. One piece of research reports on the experiences of learners aged eight to eighteen at a Computer Clubhouse located in Los Angeles (Resnick et al., 2008b). The findings of the paper demonstrated a sustained engagement with programming activities and a greater understanding of key programming concepts such as looping and conditionals.

Resnick et al.'s findings occurred in a 'computer club' environment in the US. It raises two questions: (i) whether the findings be replicated in a secondary school in Wales and (ii) whether the findings translate to a formal classroom context. WP5 of this project provides data to address the former question. WP2-4 of this thesis provides data demonstrating the efficacy of the microworlds-based approach within formal classroom settings.

2.5.3 Robotics and Playful Learning

Scratch has also been designed to move beyond the output of the screen and the inputs of the keyboard and mouse. Scratch can manage a variety of sensor inputs and motor outputs to interface with the physical world. These additional inputs are particularly relevant to the Musical Pictures microworld activity that took place as part of the pilot study in WP1. Inputs for *sound*, *light* and *slider position* were used by learners to create their own music compositions and digital artworks.

The PicoBoard (see *Figure 4*) is a piece of sensor board hardware that connects real-world sensors to projects in Scratch via a USB connection. Developed by The Playful Invention Company (2010a), the Picoboard features a sound sensor, light sensor, slider control, button control, and alligator clip connections. Using the sensor board, learners were able to measure the volume and amount of light detected by the sensors. Adjustments to the sound and images that were output by the computer could be made depending on the value of these inputs.

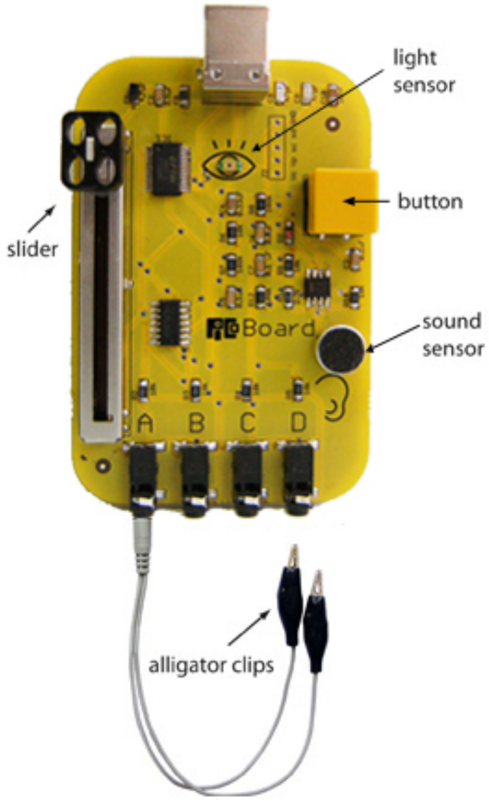


Figure 4: PicoBoard Used in WP1 (The Playful Invention Company, 2010b)

The LEGO Education WeDo construction kit has taken this link to the physical world further by combining the standard plastic LEGO brick designs with four additional 'smart' bricks (Lifelong Kindergarten Group, 2012b). The first type of brick, the hub, connects the remaining three WeDo components to the computer via a USB connection. These are comprised of a distance sensor, a tilt sensor and a motor. All of the physical WeDo components are shaped to fit together with the standard LEGO bricks so that they can form part of a physical, real-world construction.

The learning philosophy behind both the PicoBoard and WeDo construction set is something that Resnick terms *playful learning* (2004b). *Playful learning* provides learners with a set of materials, such as a PicoBoard sensor board, and enables them to learn whilst they design something they are interested in. These do not necessarily need to be physical, real-world materials. In WP5, learners were provided with a set of digital word lists and asked to generate their own poems. Playful learning occurs when learning happens *through* the processes of play and experimentation. Programming the computer to generate sentence structures and select from word lists is not an easy task. Resnick suggests that learners are more willing to persevere to find a solution to challenging programming tasks if the outcome is something that is personally meaningful, like a poem. This is something that Papert (2008b) refers to as 'hard fun'.

Resnick et al. (2000a) and The PIE (Playful Inventing and Exploring) network provided opportunities for learners to construct their own scientific instruments using programmable bricks (now discontinued) called PicoCrickets. The research showed that programming with the PicoCrickets developed scientific skills in forming hypotheses, collecting data and evaluating design (Resnick et al., 2000b). Whilst programming with PicoCrickets has produced evidence of *scientific* skill development, the research presented in this project has provided evidence that programming with the *Snap!* microworld has the potential to foster *computational thinking* and *literacy* skill development.

2.5.4 Engaging Girls with Programming

The educational benefits of programming and robotics activities have been overshadowed by the problem of inequity of access. Programming and robotics have, at least traditionally, been framed as a masculine activity. Resnick et al. (2005b) produced a National Science Foundation bid that was not funded, but their idea was to set up a series of 'Cricket Craft' clubs to engage girls with programming much earlier in their academic careers. The bid provides three recommendations that can help educators to foster engagement of girls with programming activities.

First, the bid advocates an approach that allows girls to combine programming with creative, design-oriented tasks in response to a central theme or brief. The Tanaga Challenge in WP5 was designed as a result of this recommendation for 'striking a balance' between creativity and task direction. As Resnick et al. explain:

> Twice each year, we will announce a new theme as a focus for activities at Cricket Craft Clubs. We will aim to find themes that strike a balance between being broad enough to give everyone freedom to work on a project that connects with their interests, and specific enough to foster a sense of direction and common experience (*ibid.*, p. 3).

Second, Resnick et al. warn against an overly competitive structure and instead recommend a design exhibition at the end of each project in order to provide a structure for sharing. The aim is to encourage reflection and sharing of ideas in a culture that is non-judgmental. The team compare the positive engagement of girls at a robotics event in Wellesley College, Boston with a recent LEGO League Competition where only 30% of participants are girls:

> [I]nstead of creating robots for a competition, the Wellesley students built a diverse collection of artistic and expressive creations, such as a robotic flower that closed its petals when an insect approached (*ibid.*, p.1).

Third, the team encourage storytelling and language to engage learners who preferred learning through a narrative style. The latter point, focusing on storytelling as a means of engaging girls with programming, is explored by Howland and Good (2015). Howland and Good created a programming tool that pairs block-based elements with a natural language version. The tool explains, using a textual narrative style, what will happen each time a particular sequence of blocks is used. The creators say that the unique thing about the tool is the pairing of natural language with the block-based approach: 'the pairing is a unique feature of Flip, designed to allow learners to draw upon their familiarity with natural language to "decode the code"' (p. 224). Testing for computational communication by means of a pre-test and post-test, the study revealed gains made by both girls and boys. This evidence suggests, therefore, that language provides a particularly effective context for engaging both genders with programming activities.

Gender did not figure as a guiding question in this research project, though the structure of the quantitative analyses in WP3 and WP4 meant that it was possible to analyse the impact of gender difference upon the improvements made. There was a weak correlation between gender difference and performance in the computational thinking tests reported for WP3 (43%, *see* section 6.3.1.ii) and WP4 (45%, *see* section 7.3.1.i). In the tests focusing on aspects of literacy development, a greater confidence level was reported (78%, *see* section 6.3.1.ii) and WP4 (80%, *see* section 7.3.1.i), though neither difference was statistically significant.

Three of Resnick et al.'s recommendations for engaging girls with programming were to (i) use a creative brief to structure design activities, (ii) plan for a design exhibition format and (iii) encourage a narrative focus. WP5 of this project, in comparison, (i) begins with a creative Tanaga Challenge brief, (ii) culminates with a poetry-reading exhibition and (iii) makes use of a language-focused activity. The lack of significant correlation between gender difference and performance reported in this project may potentially be a result of how the action research cycle developed in alignment with Resnick et al.'s recommendations.

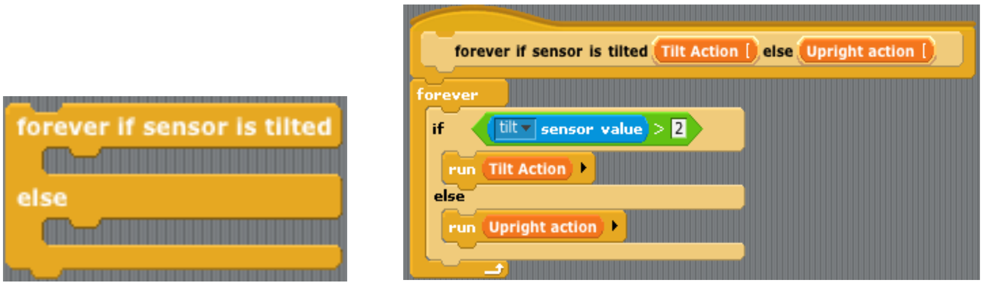
2.5.5 'Lowering the Floor' of Programming

Build Your Own Blocks (BYOB) (Mönig and Harvey, 2009) is an offshoot of Scratch that enables users to build their own blocks to supplement the standard set of Scratch blocks. BYOB has also been given a Javascript-based implementation in order to run within a standard web browser with cloud-based storage of programming projects. *Snap!* (*ibid*., 2011) is the name of the browser-based version of BYOB which, like its downloadable counterpart edition, is a block-based programming platform that is a freely available. The offline edition (BYOB) features in WP1 of this project and the online edition (Snap!) features in WP2-5.

The Snap! website provides an aim for the project: 'to extend the brilliant accessibility of Scratch to somewhat older users—in particular, non-CS-major computer science students—without becoming inaccessible to its original audience' (*ibid.*). The rationale was to improve the extensibility of Scratch by enabling the creation of a series of increasingly complex blocks for its university-level students. It aimed to do this whilst retaining the standard elements of Scratch for its younger audience.

Whilst extensibility is an important characteristic of a microworld, Rieber's (2004) taxonomy demonstrated that a 'low floor' needs to be co-present with a 'high ceiling'. There is a requirement to improve accessibility to learners at all stages of development. The empirical work of this thesis focuses on how the BYOB/Snap! platforms may be used to create a customized block kit that is targeted to specific areas of the secondary curriculum in Wales. The aim is to increase accessibility to first-time programmers whilst retaining the standard Scratch blocks to enable progression. Below we will consider an example of how this customization works.

Figure 5 shows a custom block for detecting whether or not an input sensor is tilted. This would ordinarily require a combination of four blocks (see the blocks arrangement on the right of the figure). This complex arrangement of blocks can be 'hidden' from the user at first and the functionality amalgamated into a single new custom block (see the single block on the left of the figure). When more sophisticated control of the tilt input is required, students may progress to a model similar to that of the standard scratch blocks.



```
forever if sensor is tilted
else
```

```
forever if sensor is tilted (Tilt Action [) else (Upright action [)
forever
  if <(tilt) sensor value > 2>
    run (Tilt Action) ▸
  else
    run (Upright action) ▸
```

Figure 5: Creating a Custom Block

Custom blocks can be tailored to encourage the development of ideas within a specific domain of knowledge identified by the educator. One example is *Rocket Maze;* a microworld activity trial that was used in WP1 of this project. An explanation and data file for the activity is provided in *Appendix A.* The activity requires learners to use programming techniques of increasing difficulty, such as looping and modularization, in order to guide a rocket to the moon on a simple 2D plane. This microworld takes as its focus the Welsh National Curriculum area of mathematics, in particular the angle properties of 2D shapes.

The customizations have increased accessibility relative to the original blocks but still enable the learner to experience some important computational concepts. The two blocks shown above, for example, enable learners to engage with computational thinking concepts such as variables, looping and conditionals. Designing activities with this simplified set of blocks will enable learners to construct mental models of key programming and subject-specific concepts that will become vital later on in the education cycle.

The idea that block-based programming can be simplified or tailored further beyond Scratch is gaining prominence, particularly in the US education system. We will now move to consider some of these projects.

2.5.5.i RoboBuilder

The RoboBuilder project (Weintrop et al., 2012) provides an example of what might be termed a 'component-oriented' blocks-based programming environment. Built on the OpenBlocks framework, *RoboBuilder* is a video-game environment that is designed to introduce learners to some of the key concepts entailed in computational thinking. The aim is that learners acquire such knowledge through the process of programming on-screen robots in order to defeat a number of challenging enemies.

The RoboBuilder UI (user interface) consists of two components: a programming environment for providing instructions to the robots and a battleground where the robots enter battle. Such a layout is not dissimilar to the clearly demarcated scripting pane and stage areas of the BYOB/*Snap!* UI that is used in this research project. In RoboBuilder learners use one screen to implement their robot programming strategies and a second screen for battles. The microworld activities of this project requires learners to fit together blocks in the scripting pane in order to view the impact of their actions on the stage.

A further similarity is the block-based programming approach that both research projects employ, presenting 'language primitives as on-screen objects to be assembled using a drag-and-drop interface' (*ibid.,* p. 2). This approach is advantageous to the educator because it enables them to clearly define the subject-specific boundaries of the microworld: '[the designer can] define the […] shape of each block, which in turn dictates how and where the pieces can be used' (*ibid.*, p. 3).

2.5.5.ii StratchJr

The second project, ScratchJr, is being led by Resnick and Bers (2013). The project aspires to integrate STEM learning in early childhood education through the development of a new, simplified Scratch programming language. The study also targets domain-specific knowledge in national and state frameworks including mathematics and literacy. The aims of the project are threefold:

- Support for building foundational knowledge which underlies multiple disciplines, such as sequencing, patterning, and iteration
- Support for discipline-specific knowledge from math, literacy, and classroom-selected curricula
- Support for problem-solving strategies and skills (*ibid.*, p. 3).

Even though the ScratchJr project is much larger than the modest one presented in this thesis, the parallels with the aims of this study are clear. Both projects share a common interest in making the fundamental concepts of programming more accessible to inexperienced learners. But whereas the ScratchJr project supports knowledge in the *early childhood phase* of the *US* education cycle, this research focuses on knowledge in the *KS3 stage* of the *Welsh* education cycle. The premise, however, remains the same: both projects aim to engage learners with fundamental programming concepts whilst also developing their knowledge in other curriculum areas.

2.5.5.iii Stratch 2.0

Following the success of the first iteration of Scratch, its successor the Scratch 2.0 (MIT Media Lab, 2016) launched in May 2013 with a cloud-based editor for working on Scratch projects.

Through conversation with Brian Harvey and Jens Mönig (the creators of BYOB/Snap!), new functionality was added to Scratch 2.0 to enable users to create their own custom Scratch blocks (MIT Media Lab, 2012b). The custom blocks that can be built using the Scratch platform undoubtedly provides an exciting development for educators and its relevance to this project, therefore, is clear.

Scratch 2.0 does have a number of restrictions over the BYOB/Snap! extension in terms of custom block design, however, which may make the latter platform more favourable for educators wishing to create their own microworlds. First, only command blocks can be created using Scratch 2.0. Other block types, such as the C-shape block demonstrated in *figure eight*, would not be able to be replicated using Scratch 2.0. Second, Scratch 2.0 does not allow for *block variables* to be created that are temporary and local to a particular block. Snap! essentially enables a 'hidden' variable to be created that is not accessible to learners using the variable or list monitors on the stage. Finally, later versions of Snap! provide an option to hide primitives, thereby enabling *only* the custom blocks built by the educator to be displayed. These features were important to the latter, refining WPs of this thesis in order to increase the characteristic simplicity and understandability of the microworld designs.

The custom blocks feature then, though a very welcome addition to the Scratch 2.0 platform, is not explored as part of this project as it was thought that the Snap!/BYOB platforms were more suited to the purpose here.

2.5.5.iiv Small Basic

A possible concern for educators when first introducing visual, drag-and-drop programming environments is that of how to manage the progression to text-based programming languages. The Small Basic (Microsoft Corporation, 2015) project is a text-based language that may potentially offer a user-friendly way to manage this progression. An activity created in Small Basic figures in WP3 and WP4 of this research project though no data were collected regarding how to manage this progression in programming approaches. This remains an area for future research.

With Snap!, blocks are slotted together in the scripting pane before seeing the output of programs on the stage. With Small Basic, learners type code in an area of the screen known as the editor and output their programs using either a graphics window or a text window. The Small Basic activity used in WP3-4 can be accessed using the data file in *Appendix B*.

The programming environment has been designed with a number of features in mind to increase the accessibility of text-based approaches. First, as the learner begins typing a new line of code, they meet an 'Intellisense' pop-up window providing recommendations for methods to invoke with brief explanations of each. Further, as each method is selected a more detailed explanation is provided within a pane towards the right-hand side of the screen. This provides guidance of correct syntax in addition to how this differs to similar methods.

To summarize, we began this section by looking at *Scratch* and how it popularized block-based approaches to learning programming. We then considered how this has been extended further with Snap! and the potential that this extension offers for designing customized block kits. Since two of the three guiding questions of this project focus on the potential of a microworlds-based approach for developing computational thinking and literacy, it is right that the final two sections of the literature review examines these areas of learning more closely.

2.6 Developing Computational Thinking as a Strand of Digital Competence

ICT education within the secondary phase in Wales, following its English counterpart, is currently undergoing some key changes. These need to be examined first before moving to consider where computational thinking and digital competence is situated within the current secondary curriculum in Wales.

2.6.1 Surveying the Political Landscape of ICT Education in Wales

In 2012, a report by The Royal Society examined the future of ICT in UK schools. One of its key recommendations was that ‘every child should have the opportunity to learn computing at school, including exposure to computer science as a rigorous academic discipline.’ Furthermore, the report stated that this ‘could include pupil-friendly programming environments such as Scratch, educational microcontroller kits such as PICAXE and Arduino, and robot kits such as Lego Mindstorms’ (pp. 6–9). This sentiment was echoed by the *Computing at School* working group (Bradshaw, 2012) and some key thinkers in the IT industry. Google chairman Eric Schmidt, for example, said at the time: ‘your [British] IT curriculum focuses on teaching how to use software, but gives no insight into how it's made. That is just throwing away your great computing heritage’ (Douglas, 2011).

In England Michael Gove, the incumbent Secretary of State for Education at the time, made some strong statements about the current ICT programme of study: ‘instead of children bored out of their minds being taught how to use Word or Excel [...] we can get in schools across the country [...] more programming lessons’ (Burns, 2012). As of 2014, ICT has been replaced by Computing in the National Curriculum for England. The Computing curriculum aims to ensure that all learners are digitally literate and competent users of ICT. At the core of this curriculum, however, is

Computer Science. The curriculum is designed, among other objectives, to promote programming as a fundamental skill for all learners (DfE, 2013).

In November 2012, the Welsh Assembly Government called for a review of the future of ICT and Computer Science in Wales. An ICT Steering Group was formed in January 2013 and a series of recommendations issued in September later that year. A headline recommendation calls for a new subject called Computing to replace ICT as a mandatory programme of study. The report suggests that a new statutory discipline of Computing should be disaggregated into two elements, IT and Computer Science, with the latter component encompassing programming activities (pp. 13-14).

Although there is currently no statutory requirement in Wales for students to follow a programme of study that includes programming activities, this report suggested a possible change in the near future. The report also recognised the value of computational thinking across curriculum areas and recommends that computational thinking skills should be developed from primary education onwards (p. 16).

More recently in Wales, the *Successful Futures* review by Professor Donaldson (2015) was published that focused on curriculum reform in Wales. A recommendation was issued as part of this report that Digital Competence be designated an additional area of cross-curricular development across subject areas. The Education Minister at the time accepted this, and the report's other recommendations, in full later that year (Welsh Government, 2015a).

In July 2016, a draft Digital Competence Framework was released by the Welsh Government (2016b) with the instruction to plan for its implementation in the 2016 academic year in order to have made good progress at embedding it across the curriculum by 2018. The framework was designed with the following rationale:

> The Digital Competence Framework (DCF) has been designed to encapsulate the skills that will help learners thrive in an increasingly digital world. It will improve digital competence by helping embed digital skills across the curriculum. Pupils will be required to apply digital skills to a wide range of scenarios that can be transferred to the world of work (Welsh Government, 2016a).

In a similar vein to the Literacy and Numeracy Framework (DfES, 2013), the DCF is split into a series of strands to be incorporated within all subjects when teaching five- to fourteen-year-olds. The draft strands are citizenship; interacting and collaborating; producing; and data and computational thinking. The fourth *data and computational thinking* strand of the framework is sub-divided further into elements and their constituent expectation statements for learner development. Element 4.1, entitled *problem solving and modelling*, has four expectation statements for year seven (eleven- to twelve-year olds) when beginning the secondary phase (*see* Table 3).

Further guidance is given to the form that programming activities should take in an expectation statement for year nine (thirteen- to fourteen-year olds). This clarifies that activities should take place 'in either text based or block based programming environments' (*ibid.*). The poetry activities carried out using Snap! in the empirical work of this thesis provide a block-based means of enabling learners to interact with these processes. Further, WP5 provides evidence of prediction, hypothesizing and exploration in classroom dialogue that appears to reference the third and second expectation statements quoted here.

Strand	**Element**	**Aspect**
Data and computational thinking	Problem Solving and Modelling	Identify different parts of a process e.g. variables, loops, case statements, and comments.
Data and computational thinking	Problem Solving and Modelling	Understand that changing instructions can effect or even terminate a process e.g. moving instructions around in a program could produce unexpected outcomes or cause the program to fail altogether.
Data computational thinking	Problem Solving and Modelling	Make prediction of process outcome after modifying inputs e.g. predicting the effect of changing / editing a set of instructions.

Table 3: Draft Digital Competence Framework Excerpt of Expectation Statements

This research project aims to explore the potential of a microworld-based approach for incorporating specific aspects of the literacy framework, and specific aspects of computational thinking within the digital competence framework, into cross-curricular subject domains. The value of this research, then, is particularly evident given the timely political developments within the secondary education sector in Wales.

2.6.2 Learning to Think through Learning to Program

The Welsh Government's (*2016*) draft framework for digital competence explains computational thinking as thus: 'computational thinking is a combination of scientific enquiry, problem solving and thinking skills'. More is needed here in order to explore the dimensions of this term.

Computational thinking is explained by Papert (1996) in relation to the curriculum area of mathematics: 'mathematics education can draw on computation […] to provide a much richer set of new representations of knowledge than the idea of "procedural representation" that has slipped into cognitive discourse.' Computational thinking, for Papert, is an effective method of solving problems within the domain of mathematics by drawing on the ideas of computer science.

Wing (2006) expands on this by listing some of the thinking tools from computer science that are involved in computational thinking. The ability to use decomposition when facing large problems, to modularize, to think recursively; these are all key tools to be borrowed from the computer science thinking toolkit. Wing, like Papert, suggests that computational thinking can be extended beyond the domain of computer science. Furthermore, she sees computational thinking as a tool for thought to be incorporated into *all* disciplines:

> Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science. Computational thinking includes a range of mental tools that reflect the breadth of the field of computer science […] it represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use (p.33).

A sentiment shared by Papert and Wing, then, is that the principles of computer science may be applied as a tool for thought in different subject domains - far beyond the boundaries of the discipline itself. We have already seen that Papert (1980a) makes other bold claims for the act of learning to program. He suggests that through learning to program learners also develop their high-order thinking skills. This aspect of Papert's work is particularly controversial and, perhaps, the most contested. The main opponents of Papert's work have pointed to claims that are anecdotal in nature and lacking in systematic empirical research. Pea et al. (1987) express the main thread of such criticism:

> [T]here is the expectation among many educators and psychologists that learning to program can help children develop general high-level thinking skills […] However, there is little evidence that current approaches to teaching programming bring students to the level of [...] competence needed to develop general problem-solving skills (p. 103).

Pea (1987) tracked learners who participated regularly in LOGO programming activities over the course of a year. The objects of study were the learners' programming expertise and the transferability of high-order planning skills. The study was comprised of two classes of twenty-five learners aged between eight to twelve years old: one group participated in LOGO activities whilst a second control group did not. No significant difference in planning skills between the groups were reported in the findings.

Papert (1987) responded to the findings of Pea's study by taking exception with the way in which it tested for quantifiable gains within a limited and pre-defined skillset: 'Pea and Kurland approach their experiment with a very specific idea of what cognitive effect to look for: they are checking for an improvement in a very, narrow and specific form of planning activity' (p. 10). Papert also warned that the study falls into the trap of what he terms the *technocentric* mindset. He urged against a reductive preoccupation with a particular technology whilst ignoring the wider cultural context of learning: '[technocentrism] refers to the tendency to give […] centrality to a technical object -- for example computers or LOGO' (*ibid.*).

Papert's counterargument of technocentrism may be unfairly directed towards Pea and Kurland here. Pea's major objection to Papert's work is that it paints an idealistic portrayal of Piagetian learning, expressing a concern with the removal of didacticism that such an approach promotes. Pea and Kurland (1985) make the point that 'self-guided discovery needs to be mediated with an instructional context' (p. 242). Indeed, in a different paper Pea (1984) elaborates by suggesting that, 'wherever we see children using Logo […] and learning new thinking and problem skills, it is because someone has provided guidance, support, ideas through examples, rules, and help in writing programs and discussing powerful ideas' (p. 4). Here Pea argues for a greater focus on the learning context that goes beyond the LOGO technology itself. Careful planning and scaffolding throughout the learning process is required by the teacher in order for learning gains to be made.

Pea's call for 'guidance', 'rules', 'support' (*ibid.)* need not be interpreted as a rejection of Constructionism and a championing of Instructionism. As Rieber (1992) points out, the microworld can present a compromise between Instructionist and Constructionist learning approaches. The microworld, to recite an oft-quoted idiom, is the best of *both* worlds. Instructionism calls for a curriculum of learning that is closely marked out by the teacher. Constructionism requires a more open-ended approach to learning that is characterized by a model of exploration. The microworld is able to function as a bridge between both paradigms by proposing a model of *guided discovery.* Discovery and exploration *can* still happen, but only when it is accompanied by an appropriate set of rules. Discovery learning *can* take place, but only within a constrained set of learning objectives. It is the role of the teacher, by such a model, to set these objectives: to design the rules that govern a microworld. Perhaps, then, the microworld-based approach to learning can go some way to addressing Pea's concerns.

2.6.3 Algorithmic Thinking

With careful planning by the teacher, it may be possible that a microworlds-based learning approach can develop computational thinking alongside or as part of other curriculum areas. With this in mind, a more detailed discussion of the terms *computing* and *computational* is required. *Computational* thinking, according to Peter Denning (2010), is a term derived from the previously popular *algorithmic thinking*:

> The term *computational thinking* […] was originally called *algorithmic* thinking in the 1960s by Newell, Perlis and Simon, and was widely used in the 1980s […] To think computationally is to interpret a problem as an information process and then seek to discover an algorithmic solution. It is a very powerful paradigm that has led to several Nobel Prizes (*see* ibid.).

Algorithmic thinking refers to the concept of algorithms, which makes use of four key structures (Adapted from Harel and Feldman, 2004, pp. 19–20):

1. *Direct Sequencing* – Do 'A' then proceed to do 'B';
2. *Conditional Branching* - If 'Q' do 'A', else do 'B';
3. *Iteration* – Do 'A' for a total of 'N' times or keep doing 'A' until 'Q' happens;
4. *Variables* – The value 'N', in the above, is able to change each time the program is run.

Vickers (2008) suggests that *abstraction* is an additional important skill when learning to think like a programmer. Abstraction involves thinking about things in *general*, as opposed to *concrete*, terms. According to Vickers, thinking like a programmer involves the movement from a high level of abstraction at first to a deferred lower level of abstraction later (p. 207). As Vickers suggests:

> One skill you will develop as a programmer is being able to think in terms of high-level abstractions (understanding and thinking about the problem at hand) and in terms of low level abstractions (individual data items, their format, their status) simultaneously (*ibid.*, p. 10).

Other educationalists have adopted a model of computation that moves further beyond algorithmic thinking and abstraction. Weintrop and Wilensky (2013), for example, developed their block-based programming game *Robobuilder* in order to introduce learners to the principle concepts of computational thinking. Their proposal refers to two broad computational mechanisms of 'algorithmic thinking' and 'debugging' (an experimental and cyclical process of adaptation).

2.6.4 A Model of Computational Thinking

In England, Curzon et al. (2014) from the Computing at School (CaS) Group developed a framework of computational thinking concepts. The framework makes use of five key elements suggested by Selby and Woollard (2013): algorithmic thinking (sequencing a solution); evaluation (linked with debugging); decomposition (breaking down a problem), abstraction (removing unnecessary complexity) and generalization (adapting algorithms to solve other problems). The links here with the *Robobuilder* project are clear as both formulations explicitly refer to the importance of thinking algorithmically and debugging.

Whilst the CaS framework provides a very useful way of typifying computational thinking activities, it is Resnick and Brennan (2012) who have directly drawn links between computational thinking and block-based programming environments. Taking Scratch as a context, their framework adds further detail to our understanding of how computational thinking may happen as part of a block-based programming approach. They propose a tripartite framework for describing the different dimensions of computational thinking. First, there are *computational concepts* such as *sequencing* when breaking down a program step-by-step; *conditionals* for decision making; *operators* for mathematical expressions. Second, there are *computational practices* such as *debugging* for solving problems and *modularizing* for sorting code into different stacks. Third, there are *computational perspectives* such as *questioning* (interrogating, not naturalizing technology) and *expressing* (computation for design purposes).

Resnick and Brennan's definition provides an important framework for deconstructing the computational thinking dimensions targeted by the microworld activities in this project. WP5 uses an incremental approach to the introduction of computational thinking concepts. Each tutorial in WP5 targets specific blocks that exemplify different aspects of computation rather than the entire range at once. This phased approach mirrors Howe's (1982b) implementation of a dialect of the LOGO programming language for the teaching of mathematics: 'the students' work is highly structured and involves only a small amount of program building' (p. 6).

It is also helpful to point out that microworlds can be *computer-based* and *non-computer-based*, though only the former was used in this project. Rieber (1996) reminds us that there is nothing inherent in the properties of a microworld that requires the use of a computer: 'a child's sandbox is a classic example of a natural microworld. Given buckets and shovels, the sandbox becomes a "volume and density microworld" for the child'.

We have established then that the studies shown as part of this research use Brennan and Resnick's (2012) framework is used for defining computational thinking in terms of three dimensions: computational concepts, practices and perspectives. And now that we have established what computational thinking means in this context, the final section of the literature review focuses on the relationship between computational thinking and literacy.

2.7 Exploring the Relationship Between Literacy and Computational Thinking

This research project focuses on the potential of a microworld-based approach for cross-curricular computational thinking development as a strand of digital competence. A further consideration, however, is the extent to which a microworlds-based pedagogy may also be suitable for approaching some narrower, constrained elements of literacy. As with the previous section, this section begins by looking at the current state of literacy education in Wales before moving to consider some of the issues further.

2.7.1 Surveying the Political Landscape of Literacy in Wales

In order to provide an overview of the current political landscape, first the wider context of literacy development needs to be understood. In 2003, a survey of employers throughout Wales revealed key skills gaps amongst their workforce: 'nineteen per cent of employers in Wales report skill gaps. Of those employers reporting such gaps, lack of IT skills is the most common problem, followed by communication skills' (Future Skills Wales, 2003). This concern was echoed by a review of the curriculum in Wales carried out by ACCAC (2004), the Curriculum and Assessment Authority for Wales.

The call for a more skills-focused curriculum led to the creation of a non-statutory Skills Framework for three-to-nineteen-year-olds in Wales. It was recognised that 'subject orders cannot alone adequately fulfil this requirement [for skills development] since many place an emphasis on detailed subject knowledge rather than skills development (Welsh Assembly Government, 2008i). The framework identified four skills areas; thinking, communication, ICT and number; to be developed across the different subject orders that comprise the National Curriculum in Wales.

A thematic report carried out by the inspectorate for schools in Wales, Estyn (2012), evaluated the effectiveness of the non-statutory skills framework for skills development some years later. The report revealed that the framework was largely ineffective and a key recommendation to the Welsh Government was that it 'provides a simpler structure that schools can easily use to plan, assess and track pupils' progress in developing generic skills' (*ibid.*).

In June 2011, the Welsh Government minister for Education announced the Literacy and Numeracy Framework (LNF) as a measure to improve literacy and numeracy standards in Wales. A key difference to the previous framework is expectation statements organized by year group, 'literacy and numeracy attainment matrices', to identify progression in skills development throughout a learner's development. Leighton Andrews, the incumbent Education Minister at the time, said: 'because of the *non-statutory* [*emphasis added*] nature of the Skills Framework, local authorities have given it a low priority […] we will be establishing a new National Literacy and Numeracy Framework on a *statutory* [*emphasis* added] basis' (Welsh Government, 2011).

In 2013, therefore, it became a legal requirement for all teachers of learners aged between five and fourteen to embed literacy and numeracy into all subjects. As we saw at the beginning of the preceding chapter, following the Welsh Government ratification of *Successful Futures* (Donaldson, 2015), these two cross-curricular areas of responsibility are set to be joined by digital competence - including a strand dedicated to *data and computational thinking*.

Strand	**Element**	**Aspect**	**Expectation Statement**
Writing across the curriculum	Writing accurately	Handwriting, grammar, punctuation and spelling	Use a wide range of sentence structures choosing connectives to make meaning clear
Writing across the curriculum	Organising ideas and information	Structure and organisation	Adapt structures in writing for different contexts, e.g. describe outcome, outline process or discuss an issue
Reading across the curriculum	Locating, selecting and using information	Reading strategies	Use their knowledge of: • word roots and families • grammar, sentence and whole-text structure • content and context to make sense of words, sentences and whole texts

Table 4: Literacy Framework Excerpt of Expectation Statements

The National Literacy and Numeracy Framework (DfES, 2013) sets out a set of skills that learners must develop across all subject areas. For numeracy, these skills are organized into four strands: numerical reasoning, number skills, measuring skills and data skills. For literacy, these are organized into three strands of oracy, reading and writing. As with the draft digital competence framework, each of the strands are sub-divided into elements. A further categorization layer of *aspects* is also contained in the Literacy and Numeracy Framework (*see* the excerpt in Table 4). This additional layer has been omitted from the draft Digital Competence Framework.

The three expectation statements shown in the table are directly addressed in the design of the poetry microworld activity that forms the empirical element of this research. Learners work towards the first expectation statement, the use of sentence structures, by programming each line of their poem with the grammars that are required. When learners change the lines of each poem they program, learners adapt their structure of writing in order to gain an understanding of the process of the poem's construction (expectation statement two). Finally, learners are required to populate the different word lists with examples of words that will make grammatical sense within their poems (the latter expectation statement).

The expectation statements are extremely modest in comparison to the scope of the larger literacy framework. The above list, however, is not exhaustive of all possible expectation statements that could be addressed by the microworld activities designed as part of this research. The microworld activities also required, for example, learners to think about punctuation that will be needed throughout the poems they generate. The selected expectation statements are simply the most pertinent for the microworld designs developed in this project. As with digital competence, the mapping to the literacy framework once again serves to highlight the value of this research within current education policy and practice within Wales.

2.7.2 Literacy and Language in Use

In *Cognition, Computers and Creative Writing*, Sharples (1985) describes a pilot of activities he created for teaching learners to write that is aligned with a *language in use* approach to the teaching of written English. Sharples' activities are discussed in the next section but first, time needs to be taken to consider what *language in use* actually means.

The language in use scheme for English language teaching is commonly used today in Welsh secondary schools. The scheme first requires learners to examine language use in different contexts in order to gain an understanding of its ranging applications. Learners are then required to replicate the register studied in their own writing in order to extend their range of linguistic modes (Adapted from *ibid.*, p. 51). Richards and Reppen (2014) explore the *language in use* orientation further by drawing a distinction between grammar as *knowledge* and grammar as *ability* when discussing the pedagogy of language teaching.

First, grammar is viewed as knowledge where teaching is focused on developing the rules and conventions for forming sentences accurately. Sessions of language teaching following this approach are characterized by a series of independent 'drill and practice' exercises focusing on individual grammar components. Learners are provided with exercises focusing explicitly on isolated grammar elements and grammar rules are acquired through *explicitly* learning them (*see* Ellis, 2003, p. 168).

Grammar viewed as ability, on the other hand, does not isolate individual rules of grammar and teach them 'traditionally', independently of the context of communication. Rather, grammar as ability teaches grammar that is situated within the act of communicating for a particular purpose. Rather than teaching a set of grammar points, grammar teaching is taught by providing learners with tasks for meaningful communication by asking them to construct different types of text (*see* Richards and Reppen, 2014). It is this principle that is shared with the phrase *language in use*. This is summarized by Michael Fleming at an intergovernmental conference:

> A central idea [of *language in use*] which has had a significant effect on the teaching of language as school subject is that language develops by its active use in meaningful contexts rather than just by narrow instruction in skills (Fleming, 2006).

Such an approach, however, is not without criticism. As Fleming himself asserts, critics suggest that we are now focusing too much on the *use* of language as educators whilst neglecting to reflect on the rules of grammar itself (*ibid.).* He even suggests that such a preoccupation may equal previous mistakes of focusing solely on decontextualized grammar exercises (*also see* Webb, 2016).

Where, then, does the *language in use* approach figure within definitions of what *literacy* means? Gee (1989) defines literacy as the 'mastery of or fluent control over a secondary Discourse' (p. 9). This in turn raises the question of how to define discourse, which Gee suggests is a 'sort of "identity kit" which comes complete with the appropriate costume and instructions on how to act, talk, and often write, so as to take on a particular role that others will recognize' (p. 7). What literacy is *not*, for Gee, is a decontextualized set of grammatical rules.

In 2013, a research report commissioned by the UK examination board Cambridge Assessment (2013) found that defining literacy was not a simple task. Traditional notions of literacy, such as that by Gee, have typically focused on the ability to communicate (in written and oral form) at an acceptable level. The report explains that *functional literacy* has attempted to add a requirement of *context* or *purpose* to the term: 'the level of skill in reading and writing that any individual needs in order to *cope with adult life* [*emphasis added*]' (*see* Lawton and Gordon, 1996). This certainly seems to resonate with the Welsh Government's understanding of literacy as a set of communication skills that operate within the context of *cross-curricular subjects*.

Functional literacy emphasizes the context of everyday, adult life. And the Literacy Framework by the Welsh Government focuses on cross-curricular applications of literacy. In this way, it appears that the *language in use* approach and its tenet of meaningful *context* appears to align itself with how literacy as it is understood in education in Wales today.

As a last word on literacy, it is also useful to consider a related notion that increasingly features in debates surrounding literacy and digital technologies. *New literacies* are a concept explained by Lankshear and Knobel (2013). Recalling the work of Street (1998), they explain how traditional modes of literacy are mediated by – and are being transformed by - the increasing use of digital technologies for communication. Ergodic literature presents a case in point. Hypertext fiction is characterized by branching texts where decisions made by the reader influence the development of the text itself. The ability to find meaning here resides partly in the decision-making of the reader: this is more than a basic understanding of the 'traditional views of grammar, lexicon and semantics'. As Lankshear and Knobel explain, new literacies are challenging traditional notions of literacy:

> [W]hat makes the kinds of literacy practices that are mediated and constituted by such texts "new" is partly that they depart from the conventional identification of literacy (*ibid.,* p. 5).

2.7.3 Microworlds for Language

The quintessential example of the microworlds-based learning approach is LOGO Turtle Graphics. The idea here is that, through controlling the movement of an on-screen Turtle, learners will be able to explore some key ideas in mathematics. A key example is angle of turn. Papert's original Turtle Graphics resonates in many aspects of contemporary programming or coding tutorials. Take the example of Code.org (2015). In the contemporary instead of controlling an on-screen turtle, learners are controlling Elsa from Frozen in order to make a trail of ice.

Papert is the father of the microworlds-based learning approach and set the initial domain of mathematics. There have been many contemporary translations of these ideas to block-based programming approaches, such as Code.org. Sharples (1985) applied the microworlds pedagogical orientation to the domain of written English, but these ideas have not yet been translated to block-based programming approaches and its key platforms such as Scratch, BYOB or Snap!.

In the literature review for his study, Sharples discusses the Iliad (Bates and Wilson, 1981) tutorial program for grammar teaching and is critical of its question-and-answer approach. What excites Sharples more is the debugging component of the program called the 'syntactic playground'. It is through an exploratory environment such as this one, according to Sharples, that the greatest gains in writing skills can be made:

> If this 'syntactic playground' could be adapted for use by children, it would come far closer to the aims of *Language in Use* [italics in original] than Iliad's restricted grammar drills (Sharples, 1985, p. 55).

Sharples implemented three programs in LOGO that were aligned with a philosophy of exploratory learning and the subject domain of written English, which he later extended for graphical interaction in Apple HyperCard (Sharples, 1997). He was interested in the relationship between computation and storytelling: how the computer can be used for more than merely 'just typ[ing] a story into a word processor'. His premise was clear: 'computational story writing offers new perspectives on human and digital creativity' (*ibid.*).

In 1985, one program that Sharples created was PAT: a word pattern generator. Learners first specify a structure of sentence components and a bank of different word classes. The computer, in turn, combines words and generates random phrases according to the specified sentence structures. By specifying different phrase structures and then generating sentences with random word selections, learners may observe the regularities in language and gain an understanding of its linguistic rules. Though these learning points may require careful scaffolding by the teacher (*adapted from* Sharples, 1985, pp. 63 – 64).

It stands to reason, of course, that the learner will also encounter a number of challenges along the way: English is a language that possesses many linguistic irregularities. The learner will then need to refine the rules provided to PAT even further. Or, to use a phrase familiar to computer scientists, they will need to *debug* their linguistic models. When a learner instructs PAT to create a sentence using the structure "The *ADJECTIVE NOUN VERBs"*, she may be pleased when the computer randomly generates the sentence "The grumpy man walks". Eventually, however, she may encounter a sentence such as "The pretty plane flys". The irregular movement from the base form of 'fly' to the third person singular of 'flies' may call into question this rule and debugging may be required. The teacher carefully mediates throughout the process in order to scaffold the learning that takes place.

Sharples work generated some evidence to support the idea that programming with PAT can lead to gains in written English. Using a feature analysis, Sharples was able to measure the maturity of writing features in essays that were written before and after his teaching scheme intervention. He saw an increase (though not a statistically significant one) in the mature writing features in comparison to the control group (*ibid.*). This suggests that there could be some potential impact of programming activities upon some aspects of literacy.

As Rieber (1996) explained, the guided discovery orientation that characterizes microworld-based learning approaches is not unique to using a computer. The *PLaNS* (Play Learning and Narrative Skills) project at the University of Cambridge (*see* Whitebread et al., 2015; Whitebread and Basilio, 2015) saw nine classes in the primary phase undertake a guided play approach to supporting writing skills. Learners collaborated using real-world construction tools in the form of LEGO kits. Using LEGO, learners were asked to physically construct narratives as part of their planning before progressing to write their own stories. Pre-tests and post-tests revealed significant improvements in the quality of writing for learners participating in the PLaNS project. Further, this improvement was linked to NC levels in England.

Burke and Kafai (2010) investigated how writing computer programs in Scratch can help children to develop their storytelling and creative writing abilities. Working with learners in an extracurricular setting, the study focused on the storytelling and narrative process rather than linguistic features. While 90% of learners reported in a questionnaire that their knowledge of programming had improved, 60% of learners also indicated that their storytelling had improved.

There are multiple sources of evidence suggesting, then, that language could serve as an appropriate context for microworld-based learning approaches.

2.7.4 Microworlds for Poetry

Whilst Sharples' work focused primarily on storytelling, Goldenberg and Feurzeig (1987) also approached programming from the perspective of simple poetry. Papert said that through learning to program using LOGO Turtle Graphics, the learner is able to acquire knowledge within the domain of mathematics. Sharples extended this approach to the domain of storytelling and Goldenberg and Feurzeig to poetry.

Goldenberg and Feurzeig carried out a series of projects and experiments that involve programming the computer to deal with language. An idea they share with Papert and Sharples is that learning can happen *through* programming as well as learning *to* program. Goldenberg and Feurzeig make this point clearly in their introduction to *Exploring Language with Logo:*

> By modelling language in this way, you can try out and revise your own theories as you develop them, gaining insight into both the elegance and complexity of language (*ibid.,* p. 3).

When approaching a microworld for poetry, Goldenberg and Feurzeig are first concerned mostly with its structure. What is particularly interesting is the analogies that can be drawn between the structure of a computer program and a poem. An example to illustrate is how one would go about creating a simple animation in Scratch called 'pirates'. At the procedural level, a student could use the block-based platform to create a simple animation of a boat at sea. At the sub-procedural level, this will need to be decomposed into smaller steps in order to make the waves bob up and down; another to make the crossbones flag flicker; another to make the boat sprite travel across the screen. The same, of course, is true of poetry: 'not only does the overall structure have to be modelled, but each substructure, each line in the poem, must live within its own constraints' (*ibid.*, p. 31).

The notion of *constraint* identified by Goldenberg and Feurzeig here is an important one. When constructing a poem, learners are constrained by both the semantic and syntactic rules that they have observed from previous poetry they have encountered. Sharples (1999) refers to these things as *internal* constraints in contradistinction to those that are *externally* placed upon writing activities such as task or physical resources. Sharples' PAT requires learners to think about what some of these internal constraints are as they programming the computer with different types of words to be stored in virtual 'boxes'. Logically it follows that these boxes need to be named in order to tell the computer which box to pick from: 'fly' and 'walk' are *verbs* whilst 'dog' and 'cat' are *nouns.* So the act of naming each of the boxes in turn guides the learner to reflect upon the *meta-language* or *design language* of writing (*ibid*. p. 65). Constraint, therefore, plays a crucial role in the activities that were reported.

Goldenberg and Feurzeig (1987) explored how one might go about creating a program to produce different lines of text in a poem, which is replicated below (p. 32). They use the famous 'Roses are Red' poem to illustrate.

```
TO VIOLETS
        PRINT   [ROSES ARE RED.]
        PRINT   [VIOLETS ARE BLUE.]
        PRINT   PRAISE
        PRINT   RHYMOO
END


TO RHYMOO
        OUTPUT (   SENTENCE   [AND]   WHO MOO   )
END

TO MOO
        OUTPUT PICK [   [DOES TOO]   [LOVES YOU]   [SNIFFS GLUE]   [SIPS DEW] ]
END
```

The program here is made up of three different procedures. The procedure *VIOLET* is used to create the overall structure of the poem. We can see that this simply prints four lines. Notice that the first two lines will always remain the same: these will always say 'roses are red' followed by 'violets are blue'. The third line refers to a procedure called *PRAISE* that we do not see here.

PRAISE is where a compliment will be paid to the poem's recipient on Valentine's Day. The final line of *VIOLETS* calls *RHYMOO. RHYMOO* is used to produce a sentence with a random '-oo' sound ending each time. This is achieved in two stages. First, a sentence beginning is provided by *RYHMOO*. Second, *MOO* randomly picks a sentence end from, in this case, a choice of four: 'does too', 'loves you', 'sniffs glue' or 'sips due'.

The mechanism above is important because it enables poems to be created and analysed using multiple levels of abstraction. Indeed, Wing (2006) and Vickers (2008) reminded us in an earlier section that this is an important part of thinking computationally. Learners create a set of rules to define the poetry form and create more rules to construct the lines within that form. Here we have seen an example that focuses on a rhyming poetry forms but, as Goldenberg and Feurzeig themselves recognise, 'sometimes, as with haiku, one may restrict the number of syllables that may be used' (*ibid*., pp 33 – 34).

The work by Sharples and Goldenberg and Feurzeig demonstrates how educators, through harnessing the power of existing computational tools, could construct microworlds to facilitate knowledge acquisition within subject domains focused on literacy. This notion maps directly to the empirical work of this thesis. Whereas Goldenberg and Feurzig use LOGO in the 1980s to design microworld activities for the subject domain of written English: this thesis uses Snap!/BYOB in a contemporary educational context to design a new, block-based poetry microworld with a particular focus on specific aspects of literacy and computational thinking development.

2.7.5 Poets and Programmers

This literature review has traced the microworlds-based learning approach from Papert's Turtle Graphics in the 1980s through to block-based languages such as *Scratch* in the modern day. The novelty of this thesis lies with the idea that the extensible nature of contemporary block-based programming environments, such as BYOB and Snap!, provides educators with an opportunity to create their own customized toolkits for exploratory learning within subject domains. Taking Sharples and Goldenberg and Feurzeig as inspiration, much of the project focuses on how far microworlds for simple poetry can develop specific aspects of literacy and computational thinking.

But this object of enquiry is set against an assumption that programming is a suitable vehicle for poetry. It is pertinent to end, then, with thoughts on what it is about making poetry that is particularly suited to the act of making a computer program.

One possible answer to this question is provided by the software engineer Fred Brooks who, in 1974 (later republished in 1995), observed that:

> The programmer, like the poet, works only slightly removed from pure thought-stuff. He builds his castles in the air, from air, creating by exertion of the imagination. Few media of creation are so flexible, so easy to polish and rework, so readily capable of realizing grand conceptual structures (p. 7).

Brooks' linking of imagination and programming here appears to map to John Maeda's linking of aesthetics and computation in *Creative Code* (2014). A suggestion shared by both writers is that coding can operate as a creative medium. This is an assertion that has spawned further research, such as that carried out by Kafai and Peppler (2005), into creative coding projects. If one subscribes to the view that programming is an activity partly defined by creativity, one could postulate a link between the creative processes involved in poetry and programming. The shared creativity of writing a poem and coding a computer program may makes poetry a particularly appropriate context for learning about programming.

An alternative view that may work to suggest a close relationship between poetry and programming can be posited by extending the work of Vee (2013). Vee begins by moving beyond popular constructions of programming as a 'fourth literacy'. Instead Vee suggests that coding a computer program is actually itself part of textual literacy and not entirely distinct from it. Vee defines literacy as 'as a human facility with a symbolic and infrastructural technology—such as a textual writing system—that can be used for creative, communicative and rhetorical purposes'. According to Vee, this is something that literacy shares with computer programming: 'Like textual literacy, computer programming is also a human facility with a symbolic technology— code—that allows people to represent and interpret ideas at a distance' (p. 45).

Writing a computer program, according to Vee, may be a particularly effective context for learning about poetry (and vice versa) because the act of programming is not entirely distinct from the act of writing. The similarities in the skillset employed when writing a computer program and writing a poem mean that the two activities work particularly well together and complement each other. This contrasts with the former construction by Kafai and Peppler (2005) that instead points to a related base of creative skills. Shared by both models of what it means to *write* and what is means to *code*, however, is the notion of considerable overlap between the two skill sets. In this sense, both writers reach a similar conclusion though their arguments are framed differently.

Perhaps, as Vee argues, as programming evolves the line between text and technology is increasingly blurring with highly readable yet sophisticated programming languages being developed. She argues that we are currently moving towards a model of 'programming as literacy' where 'programming is not replacing writing, but is rather interlacing with it, augmenting it' (p. 59).

3 Methodological Overview

This project uses a mixed methods approach that combines quantitative and qualitative instruments across five work packages (WP). The guiding questions for the research were formed following the initial pilot study carried out in WP1. Whilst the individual chapters detail specific methodological foci, this section is intended to provide an overview.

The design-based research paradigm 'aims to improve educational practices through iterative analysis, design, development and implementation' (Wang and Hannafin, 2005, p. 6) and so seems initially to be a suitable choice of research approach given the guiding questions of the project. All but one of the WPs (WP2) reported here, however, takes place at an organisation where the teacher-researcher is employed as a teacher of ICT. Further, the project is specifically intended to bring about an improvement in teaching and learning within one particular organisation. The dual role of the teacher and researcher within a school that is familiar meant that action research was the most suitable choice for the project. This research paradigm will now be considered in more depth.

3.1 The Action Research Approach

Punch (2005) reminds the reader that 'a good way to achieve a fit between questions and methods is to ensure that the methods we use follow from the questions we seek to answer' (p. 20). Janesick (1994) uses term *methodolatry,* a portmanteau of *method* and *idolatry,* to describe research studies where this sequence is detrimentally reversed: 'a preoccupation with selecting and defending methods to the exclusion of the actual substance of the story being told' (p. 215).

The research paradigm adopted for this study may fall broadly into the category of what has become known as *action research*. A little time needs to be spent unpicking what is meant by such a term. The aim is follow Punch's advice and demonstrate the appropriateness of action research to the guiding questions of the study.

Cohen, Manion and Morrison (2011) provide a very clear formulation of what they consider to be the 'prime feature' of the action research paradigm:

> Action research [...] is a powerful tool for change and improvement at the local level [...] the desire for improvement to practice, based on a rigorous evidential trail of data and research (*ibid.*, Locations 15240-15267 of 33832).

Making use of a mix of quantitative and qualitative methods, the project seeks to establish whether improvements in specific aspects of computational thinking and literacy development follow from incorporating elements of microworld-based pedagogy into teaching practice. This investigation was strongly localized to the organization where the teacher-researcher is employed as a teacher.

A key shortcoming of the action research model, however, is the restricted generalizability that it provides. Whilst generalizability is supported to an extent with the setting of WP2 at a different school, there were two key benefits to the action research model.

First, action research is a reactive and adaptive methodology. The research questions were finalized following the initial pilot in WP1. Whilst the research questions did not change from this point onwards, some key methodological changes were made following each stage of the research. As Denscombe (1998) explains:

> [Action research is] a feedback loop in which initial findings generate possibilities for changes which are then implemented and evaluated as a prelude to further investigation (p. 58).

This notion of a 'feedback loop', allowing for a continual methodological refinement based on the findings, was essential to the project. WP1 started by exploring, in a very general and 'broad stroke' way, the microworld-based pedagogical approach across many distinct curriculum areas. These findings then led to a particular focus on cross-curricular literacy development in the following WPs. The adaptability of the action research approach, as Cohen and Manion (1994) explain, has a great deal to offer classroom-based projects such as this one:

> A feature which makes action research a very suitable procedure for work in classrooms and schools […] is its flexibility and adaptability (p. 192).

Second, action research is beneficial because the contextualized nature of this approach aligns with the aim of the project to bring about improved practice within a single school:

> [Action research aims] to arrive at recommendations for good practice that will tackle a problem or enhance the performance of the organization (Denscombe, 2002, p. 174).
>
> Action research is appropriate whenever specific knowledge is required for a specific problem in a specific situation (Cohen and Manion, 1994, p. 194).

The action research model is sometimes referred to as a 'teacher as researcher' model of research. This phrase presents the dual role for the teacher/researcher. As a teacher, he/she is concerned with teaching practice and pedagogical development. As a researcher, he/she is concerned with the improvement and/or problem solving of issues that arise from practice. As Johnson explains:

> Action research is deliberate, solution-oriented investigation that is […] characterized by spiraling cycles of problem identification, systematic data collection, reflection, analysis, data-driven action taken, and, finally, problem redefinition. The linking of the terms

> "action" and "research" highlights the essential features of this method: trying out ideas in practice as a means of increasing knowledge about and/or improving curriculum, teaching, and learning (Johnson, 1993).

In an oft-quoted paper, Lewin (1946) describes action research as 'a spiral of steps, each of which is composed of a circle of planning, action and fact-finding about the result of the action' (p. 206). The cyclical approach enabled the methodological design of each intervention in the project to respond to the prior findings. For example, in WP4 a pair programming approach and more teacher-directed activities were used following the survey data collected in WP3.

3.2 A Mixed-methods Approach

The aims of this project are threefold. First, to examine the potential of a microworlds-based pedagogy for bringing about improvements in computational thinking. Second, to see if this potential also extends to defined aspects of literacy development. Third, to establish recommendations for educators wishing to replicate elements of a microworlds-based pedagogy in their own teaching practice. Whilst the research questions demonstrate a clear need for quantitative analysis of outcomes, qualitative insights are also needed. Bell (1993) provides a clear distinction between quantitative and qualitative paradigms of research within education:

> Quantitative researchers collect facts and study the relationship of one set of facts to another. They measure, using scientific techniques, that are likely to produce quantified, and, if possible, generalizable conclusions. Researchers adopting a qualitative perspective are more concerned to understand individuals' perceptions of the world. They seek insight rather than statistical analysis (pp. 5 – 6).

What is clear from Bell's distinction is that, in order to address all three question, elements pertaining to both paradigms are necessary. A quantitative element is required in order to attempt to establish causality between the microworld-based interventions and gains in outcomes. Such approaches are limited in their ability to account for the complexity of human response (see Hughes, 1997) and as such a qualitative element is needed. Johnson and Onwuegbuzie (2004) make a clear case for using a mixed methods approach in educational settings:

> We hope the field will move beyond quantitative versus qualitative research arguments because, as recognized by mixed methods research, both quantitative and qualitative research are important and useful. The goal of mixed methods research is not to replace either of these approaches but rather to draw from the strengths and minimize the weaknesses of both in single research and across studies (pp. 14-15).

Creswell's (2014) discussion of research design provides further justification for the mixed-methods approach taken in this project. Quantitative data, derived from quasi-experiments, 'seeks to determine if a specific treatment influences an outcome' (p.42). In WP2-4, t-tests were used to establish performance gains in defined aspects of literacy and computational thinking. Qualitative data derived from observation and learning journals were also required for an 'open-ended' analysis in order to form the GQ3 effective practice recommendations (*ibid.)*. The two types of data were collected at the same time in each of the work packages and collated to form the conclusions. This approach is described by Creswell as convergent parallel mixed methods:

> Convergent parallel mixed methods is a form of mixed methods design in which the researcher converges or merges quantitative and qualitative data in order to provide a comprehensive analysis of the research problem (*ibid.,* p. 44).

The design of this project aligns with Creswell's approach with respect to convergent parallel mixed methods.

3.2 Instruments

As part of the mixed-methods approach, continually adapted as part of the action research cycle, both quantitative and qualitative instruments were used for data collection.

3.2.1 Quantitative Instruments

In order to assess improvement in literacy and computational thinking, a true experimental design (see McGowan, 2011) would strengthen claims for quantitative gains following the microworld-based intervention. Such a design would make use of a randomized sample of learners consisting of a control group and an experimental group. The experimental group would receive the microworld-based intervention whilst as many other factors as possible are controlled for. Using a pre-test and post-test, there is then a strong case for causality following the intervention.

The quantitative element of this project, however, makes use instead of a quasi-experimental design (see Muijs, 2011). A quasi-experimental design meant that classes could be selected that were broadly of the same attainment level and gender balance. They were selected using a nonprobability strategy according to timetabling constraints and the willingness of subjects to participate. Therefore a comparison group was used in lieu of a control group and participants were taught in their regular classes during regular scheduled lessons. Whereas the experimental group created poetry using the microworld activities, the comparison group used offline methods. In WP4 for instance (*see* section 7.2), the comparison group scripted haikus in small groups using flipchart paper that were then performed to the rest of the class. Despite the problem of transferability of findings, the experimental-comparison group approach meant that research findings were firmly embedded within a realistic classroom setting.

There are two justifications for this. First, the 'lab based' environment required by an experimental design would work to distance the research from realistic school-based practice.

This clearly runs counter to the principle of embedded research that action research promotes. Second, such a step would require a reorganization of the timetable and impact upon staffing of classes not participating in the research. Teachers experiment and refine their pedagogical approach with different classes as part of their day-to-day teaching practice: the research presented here does not extend beyond this and no timetable changes or reorganization of classes were required.

> Quasi-experimental designs are meant to approximate as closely as possible the advances of true experimental designs, where the problems […] such as having to implement a programme in a natural school setting, occur (*ibid.,* p.23).

WP2 and WP4 make use of this quasi-experimental element using a pre-test/post-test design (see Dimitrov and Rumhill, 2003). Learners in both the comparison and experimental groups were provided with two pre-tests to assess their knowledge in specific aspects of computational thinking and literacy. The experimental group then undertook a pedagogical intervention involving computer-based microworld activities whilst the comparison group undertook activities using standard pedagogy. Two post-tests were provided following the intervention to measure any difference in gain following the intervention with the experimental group. Both the intervention and comparison groups within each WP were taught by the same member of teaching staff. Using the same teacher for both groups was a step taken to reduce the effect of differences in teaching characteristics upon learning outcomes.

When testing for gains in written English following a computer-based intervention, it is useful to consider work undertaken by Sharples (1985). When commenting on the design, he specifically states that he is interested in 'the learning *process*' and is reluctant to focus merely on 'measures of *learning outcome'* [italics in original] (*ibid.*, p. 86). Nevertheless, he does make an attempt to quantify the 'transfer of skills from the teaching scheme' (p. 87). He achieved this by asking all learners to produce two essays, one before the start of the activities and one at the end. This enabled a comparison of the frequency of mature writing features to be made.

A related study focused on mathematics by Howe (1984). Here learners were issued with a short test *before*, and a test *after,* the microworlds-based teaching intervention. This performance measure was used by Howe to assess mathematical changes in learners following the implementation of a dialect of the LOGO programming language. Howe comments that this methodology was useful for 'assessing if and how a child's understanding of a problem domain changes while he is working in these [LOGO programming] environments' (p.2).

Of these two models of assessing changes following the intervention, the pre-test/post-test method used in WP2-4 of this research project more closely mirrors the method by Howe. Both the assessments and the microworld activities were designed by the researcher in conjunction with subject specialist teachers.

3.2.2 Observation

Observation was a significant method of data collection used in this project in order to collect qualitative data that contributed towards all three GQ. Studies relying on observational data such as this broadly fall into one of two types: *participant* and *non-participant* (see Bell, 2005). This study uses the former type of observation, described by Lacey (1976) as thus:

> [T]he transfer of the whole person into an imaginative and emotional experience in which the fieldworker learned to live in and understand the new world (p. 65).

The researcher is employed as a full-time ICT teacher at the institution where four of the WPs were carried out, and so a participatory mode of observation was the unavoidable choice. As the researcher-participant is currently deeply immersed within school life through his role as a teacher, non-participatory observation would be difficult to achieve given he is known to the learners and staff of the organisation. Bell (1993) expounds further on the part that participant observation has to play in a research study:

> Most studies of this kind are largely unstructured. That is, the researchers do not start with preconceived ideas about precisely what is they want to observe. They observe events, situations, behavior and then write-up all their observations immediately afterwards (p. 110).

The unstructured approach of participant observation is open to criticism. Not least there is the issue of bias and subjectivity that arises when attempting to report objectively on an organization that one is part of. Added to this is the issue of familiarity: a lack of objective 'distance' can cause certain aspects of the data to be overlooked (*ibid.*). Familiarity was a particular concern in WP1 and WP5 where the microworld activities were led by the researcher as part of his standard teaching timetable or an extracurricular club. Familiarity was a lesser issue in the other WPs as the researcher assumed a technical support role within classes taught by different teachers.

A further concern raised by this instrument is inaccuracy caused by delays in the write-up of observation notes. As Denscombe (1998) explains: 'it is so easy to forget things [...] if field notes are delayed' (p. 151). To attempt to minimize such inaccuracies, the researcher followed the advice of Bell (1993): '[field notes] are written up as soon as possible after the observation' (pp. 110 – 111). Furthermore, a section of this methodology (*see* section 3.3) is dedicated to strategies for improving the validity and reliability of the data collected.

WP4 and WP5 reduced the possibility of such inaccuracies by making use of a series of voice recordings when learners were undertaking the pair programming activities. These were transcribed following the intervention. Audio recordings provided direct quotations that were useful when writing up the findings. As Bell (1993) elucidates:

> Tape recordings can be useful to check the wording of any statement you might wish to quote and to check that your notes are accurate. They can also be useful if you are attempting some form of content analysis and need to be able to listen several times in order to identify categories (p. 96).

3.2.3 Survey Data

Surveys (see *ibid.*) documented the learning that had taken place following the interventions in WP1 and WP3. In WP1 a Moodle database (*see* section 4.2.1) was used and in the latter WP (*see* section 6.2.2) this took the form of an online survey. Each learner tracked the activities they participated in, and reflected on the learning that had taken place, following the microworld-based interventions. The entries were valuable in that they provided a dataset to supplement the observational data, and were necessary for 'background information and understanding of issues that would not otherwise be available' (Hopkins, 2002, p. 78).

As Bell cautions, however, it is important when using this method that 'instructions need to be explicit' (Bell, 1993, p. 103). Care was taken, therefore, to sufficiently guide learners on what content to include in their responses. A structured sequence of text areas and multiple-choice checkboxes were provided to guide responses sufficiently. Learners were asked to comment on the nature of the activities they had undertaken, what learning they felt had taken place and how the activities could be improved. Learners were provided with sufficient time to update their response at the end of the session and this was well planned for as part of the teaching schemes from the outset.

3.3 Maximizing Credibility

As previously stated, the researcher in this project also works as a full-time teacher at the establishment that provides a context for the research. The issue of familiarity is important here: there is a requirement, as Delmont (2002) puts it, to render 'the familiar strange' (p.15).

In an attempt to minimize issues that may impact on credibility, this study has been designed with Denzin's (1978) taxonomy of triangulation in mind. Triangulation is defined by Cohen and Manion (1994) as 'an attempt to […] explain more fully […] the richness and complexity of human behaviour by studying it from more than one standpoint' (p. 233). Denzin's four modes of triangulation will now be considered in turn and related to the design of this study.

3.3.1 Data Triangulation

The first method of triangulation that Denzin identifies is multifaceted in its approach. For Denzin, the data form of triangulation may come in one of three sub-types: time, space and person. Data triangulation may be achieved by varying *when* (time triangulation), where (space triangulation) and from whom (person triangulation) data is collected. Hussein (2009) explains these three sub-types clearly:

> According to (Denzin, 1978), there are three types of data triangulation; namely, time, space and person. These types of data triangulation come as the result of the idea that the robustness of data can vary based on the time data were collected, people involved in the data collection process and the setting from which the data were collected (p. 3).

This study achieves space triangulation by situating WP2 in a different school and person triangulation by using different samples of respondents throughout the WPs. Cohen and Manion (1994) differentiate between two types of what may be termed *time* triangulation:

> Time triangulation [makes use of] cross-sectional and longitudinal approaches. […] Cross-sectional studies collect data concerned with time-related processes from different groups at one point in time; longitudinal studies collect data from the same group at different points in the time sequence (p. 236).

This study attempts to achieve time triangulation by adopting the former, *cross-sectional,* approach. The latter three WPs compared the difference in learning outcomes of samples from the school populations at a given point in time.

3.3.2 Investigator Triangulation

The difficulty with studies such as this one that are partially reliant on observational data, as Cohen and Manion point out (1994, p. 238), is that different researchers have different observational styles. This will be reflected in the data that is collected, irrespective of how much thought is given to minimizing bias in the design stages of a study. The intervention that took place as part of WP3 was viewed for one hour by two academics from the University of South Wales. One academic viewed WP5 for five hours. By discussing the findings with different observers, an attempt to achieve investigator triangulation was made. The benefit of this *inter-judge* approach is explained by Black (2002):

> This [inter-judge approach] is highly appropriate for such activities where personal judgment is involved, for such situations that require checking the consistency of observers (p.86).

It is also worth noting here that the field notes made by the observer were subjected to respondent validation before being included here. In WPs led by subject-specialist teachers, the teachers themselves were asked whether or not they agreed with the findings. In the work packages that were not led by subject specialists, the findings were discussed with the visiting academics. This approach is advocated by McCormick and James (1983): validation is achieved when [...] subjects of the research, recognize its authenticity (p. 241).

3.3.3 Theoretical Triangulation

A further problem that can impact on the credibility of a study is its theoretical basis. There is sometimes a tendency to focus on one viewpoint without paying due consideration to alternatives:

> Researchers are sometimes taken to task for their rigid adherence to one particular theory or theoretical orientation to the exclusion of competing theories (Cohen and Manion, 1994, p. 237).

This research focuses in particular on a microworlds approach to learning and much of the literature considered, therefore, adopts a Papertian perspective. The literature review of this project did attempt, however, to contrast this perspective with that of other theorists in this area. In particular, the review considered the alternative theories of learning proposed by Piaget and Vygotsky in light of Papert's methods. There was also coverage given to the criticisms directed towards Papert's work by Pea et. al. (1984; 1987), Simmons et al. (1993) and Ennis (1992).

3.3.4 Methodological Triangulation

The final type of triangulation, that which is methodologically based, is defined by the Open University in a course study guide:

> Cross-checking the existence of certain phenomena and the veracity of individual accounts by gathering data from […] a number of sources and subsequently comparing and contrasting one account with another in order to produce as full and balanced a study as possible (Open University course E811, 1998, p. 54).

In line with such a model, this study attempts to achieve methodological triangulation by using different research instruments (survey data, observation, gains in learning outcomes) to investigate the same research questions. In the words of Cohen and Manion (1994), 'different methods' were used on the 'same object of study' (p. 236).

In Denzin's taxonomy (1978), two sub-types of methodological triangulation are identified. *Within methods* triangulation repeats a study to check credibility whereas *between methods* triangulation uses multiple instruments on the same object of study. This project makes use of both sub-types. The work package format enabled earlier sub-studies to be replicated and refined whilst there were also three different research instruments used. This approach enabled the researcher to 'search for regularities in the research data' (O'Donoghue and Punch, 2003, p. 78).

3.4 Data Analyses

3.4.1 Qualitative Analyses

Two different analyses were carried out for the qualitative data collected across the five WPs.

3.4.1.i Content Analysis of Survey Data in WP3

The qualitative data derived from the surveys in WP3 were analysed using an open coding method. Such an approach, inherited from a grounded theory model of content analysis, is characterized by findings that are derived directly from the data collected. Following the content analysis, a coding frame was established in order to write up the qualitative narrative of WP3 (*see* section 6.3.2). As Strauss (1989) puts its, this requires the researcher to make 'analytic abstractions that are […] grounded on close inspection of the data' (p.58). Strauss and Corbin elucidate further (1998):

> A researcher does not begin a project with a preconceived theory in mind […] Rather the researcher begins with an area of study and allows the theory to emerge from the data […] Grounded theories, because they are drawn from the data, are likely to offer insight, enhance understanding, and provide a meaningful guide to action (p. 12).

A potential pitfall with this method is that the researcher must consider inferences or implications of responses in order to establish the coding frame (see Hancock, 1998, p. 21). A further issue is the possibility of researcher preconceptions that justifies the need for strong triangulation. The subject specialist teacher at the school in WP3 was available to check the write-up of the findings. This enabled the subject specialist teacher to function as a corroborative measure of the study (*see* Berg, 2001, pp. 242 – 243).

3.4.1.ii Discourse Analysis of Voice Recording Data in WP4 and WP5

The second mode of qualitative analysis used does not follow a grounded theory approach as the findings were structured independently from the data itself. In order to analyse the voice recording data of pair programming in WP4, a simple pre-defined coding scheme was used according to Wegerif and Mercer's (1997) framework for researching peer talk in the classroom. Talk sequences were categorized according to whether they were (i) disputational, (ii) cumulative or (iii) exploratory and their narratives analysed in light of these categorizations. The framework for peer talk analysis is discussed in further detail in WP4 (*see* section 7.2). The framework was useful for providing an overview peer talk analysis: the limited number of three talk categories enabled a high-level, broad-grained analysis to take place.

For WP5, a more comprehensive coding scheme was used for analyzing the paired programming dialogue. This thus enabled a more finer-grained examination of the learner dialogue that took place. Hennessy et al. (2016a) - building on the work of Wegerif and Mercer, and many other theorists in educational dialogue - provide a methodological scheme for coding the dialogic interactions when undertaking the microworld activities. The SEDA (Scheme for Educational Dialogue Analysis) scheme (*ibid.*) was used for this study to maximize granularity in this final dialogue analysis. The Mercer and Wegerif (1997) framework, though useful for providing an overview, characterizes communicative acts according to three types of sequence only. The SEDA scheme widens this to thirty-three communicative acts grouped within eight clusters. The 'mixing' of instruments and coding frames in this way is a flexibility offered by the convergent parallel approach (*see* Creswell, 2014).

3.4.2 Quantitative Analyses

In addition to the content analysis of qualitative data, quantitative methods were used to analyse the subject knowledge assessments that took place before-and-after each microworld activity in WP2, WP3 and WP4. A simple mark scheme was devised in conjunction with subject teachers for these assessments. By comparing the difference between the *before* score and the *after* score for each participant, a *gain* score was produced. The resulting positive or negative residual provided a means to measure the impact of the microworld activity upon the aspects of literacy and computational thinking that were tested for.

Data analyses were carried out using IBM SPSS in three stages.

Initially, a univariate analysis was completed using the mean as a measure of central tendency. The mean gain between the pre-test and post-test computational thinking scores was calculated for the experimental group. Next, the mean gain between the pre-test and post-test literacy scores was calculated for the experimental group. Both analyses were repeated for the comparison group.

Following this, a bivariate analysis was used to compare the performance of both groups. The relationship between the independent variable *group* (experimental or comparison) and the dependent *gain* (post-test minus pre-test score) was investigated for both the literacy and computational thinking datasets. Independent samples t-tests were carried out to look for *statistical* significance whilst recognizing the difficulty of achieving this with a small sample size (see Muijs, 2011). Additionally, Cohen's D was calculated as a measure of effect size to determine levels of *practical* significance (*see ibid*.) following the microworld intervention.

It is important to note that the suitability of the t-test as an instrument can be questioned here since the samples violated the assumption of random assignment from the school population. The quasi-experimental design, though useful for gathering results within a classroom context, did not allow for random sampling. The t-test was carried out, however, as there is evidence to suggest that the t-test can be quite robust despite such violations (see *ibid.*). It is also important to reiterate that the small sample size makes achieving *statistical* significance difficult. It is for this reason that the effect size calculations were also carried out to investigate *practical* significance.

Finally, a multivariate analysis made use of a regression model to investigate the relationship between two predictor variables (i) experimental group and (ii) computational thinking gain upon the dependent literacy gain. A multiple linear regression model was used to examine whether (i) being part of the microworld intervention group and (ii) scoring highly in the computational thinking tests, when *combined*, could effectively predict a higher score in the literacy tests. In short, the aim was to explore the combined effect of high scores in computational thinking and being part of the intervention group upon performance in the literacy test.

3.5 Sample

3.5.1 School 1 in Context (WP1, 3, 4, 5)

Time needs to be spent discussing the institutional context of this research in order consider how its findings may transfer to schools situated within a similar setting. As this project uses an action research method, learner samples from School 1 feature in all but one (WP2) of the five WPs in this project.

In Wales, all schools are required to be periodically inspected by Estyn, Her Majesty's Inspectorate for Education and Training and Wales, under section ten of the 1996 Schools Inspection Act. The contextual data here summarizes a recent report into the quality and standards of provision at this education provider (Estyn, 2016).

School 1 is a large English-medium secondary school that is maintained by a Unitary Authority in the South East region of Wales, UK. It is a large school, with 1,521 mixed-gender learners on roll between the ages of 11 - 18.

The school admits students with a varying range of abilities. In Wales, students experiencing learning difficulties are placed on a register of Special Educational Needs (SEN). The percentile of compulsory school age learners appearing on the SEN register for School 1 was 17% at the time of the inspection. This is nearly half that of the National comparator of 25.1%.

In Wales, the percentage of learners entitled to free school meals (FSM) serves as a measure of social and economic deprivation within a school's intake. The FSM measure for School 1 is 6% of students on roll. This is significantly lower than the National comparator of 17.4%. A majority of learners on roll come from geographical areas of socioeconomic advantage.

Both primary and secondary phases of the Welsh education system employ a common system of numbering academic year groups according to learner age. Academic year groups are further combined to form four key stages (KS) of development. In secondary schools, academic years seven to nine form KS3 and consists of learners between the ages of 11 – 14. KS4 is made up of years ten to eleven and consists of learners between the ages of 14 – 16. KS4 marks the end of compulsory education and normally culminates with the learner achieving a portfolio of GCSE or equivalent qualifications. Many schools, including School 1, also offer a 'sixth form' where learners aged 16 – 18 may continue to study a portfolio of A Level or equivalent qualifications. 16 – 18 education is often referred to as 'KS5', though technically the KS designation ends at compulsory school leaving age.

In Wales, there are many indicators of academic progress used to measure performance of 11 – 18 education providers. Here, we consider one measure in KS3 and two in KS4.

At the end of KS3, a core subject indicator is used to assess performance in English (Welsh in Welsh-medium schools), mathematics and science at the expected level 5 or above. In 2016, the school achieved benchmark quartile 4 and is as such in the lowest performing 25% of schools with similar levels of FSM to this school.

At the end of KS4, a threshold level 2 inclusive indicator is used to measure the level of learning equivalent to five A*- C GCSEs (including mathematics and English) that has taken place. In 2015, the school achieved benchmark quartile 3 and is as such in the lowest performing 50% of schools with similar levels of FSM to this school.

At the end of KS4, a level 2 indicator is used to measure the level of learning equivalent to five A*-C GCSEs that has taken place. In 2015, the school achieved benchmark quartile 4 and is as such in the lowest performing 25% of schools with similar levels of FSM to this school.

Two out of three of these indicators are well above the national average for Wales, though performance is in the lower two quartiles when this performance is compared against schools with similar levels of eFSM. The most recent inspection report includes five recommendations to be addressed in school improvement planning. Recommendation 3 states that the school should 'improve the quality of teaching, including supporting the development of pupils' literacy and numeracy skills' (ibid., p. 4).

In WP1, the microworld-intervention took place within discrete ICT lessons at School 1. The researcher is a teacher of ICT at School 1. As part of this role, the researcher is directly responsible for the delivery of School 1's KS3 discrete ICT programme of study. The curriculum at KS3 is designed to address the requirements of the *Information and Communication Technology in the National Curriculum for Wales* document (DCELLS, 2008b). The KS3 curriculum is divided into eighteen six-week units where learners study six units per academic year. This structure maps loosely onto the structure of the academic year within the Welsh education system. Learners study a broad range of applications including modelling with spreadsheets; data handling with databases; communicating information for interactive and print-based media. Learners receive one-hour of discrete ICT tuition per weekly timetable cycle.

In WP3 and WP4, in line with a focus on cross-curricular literacy, learners undertook activities led by a subject-specialist English and drama teacher. The choice was made to trial these activities within drama lessons as opposed to English lessons. This provided the opportunity to address a more diverse sample in terms of academic attainment since, unlike English but like ICT, drama is a non-streamed subject area at the school. Again learners receive a single hour of drama per week in KS3. Classes are taught with around a 30:1 learner-teacher ratio and are led entirely by subject-specialist teaching staff.

In WP5, an extracurricular club led by the teacher-researcher was the setting for the discourse study. Data was collected over the course of two half terms (around twelve weeks) at the club. Attendance was variable but, generally speaking, a core of around five to six participants attended most weeks with some weeks attracting up to around fifteen students. The club took place during lunchtime for around forty-five minutes per weekly session.

3.5.2 School 2 in Context (WP2)

The most relevant Estyn (2010) inspection report for School Two at the time of WP2 provides an indication of the context of the school population from which the sample was drawn.

The school is a large school with over 1,500 learners on roll at the date of the last inspection. Using free school meals (FSM) as an indicator of socioeconomic disadvantage, in 2009/10 eligibility was 36.8%. This was significantly greater than the national comparator of 17.1%. This is also significantly greater than School 1's figure. It is clear to see, then, that School 2 admits a much greater proportion of students who are living in socioeconomic disadvantage in comparison to that found in School 1.

A high number of learners (68%) were from homes where languages are spoken other than English or Welsh. As such, there was a much higher level of English as an additional language

(EAL) support provision in place at School 2 in comparison to School 1. The percentage of learners appearing on the register for special educational needs (SEN) is 28.6%. This was significantly higher than the national comparator of 20.9% and School 1's figure. There is, then, a clear difference in context evidenced by the inspection data gathered for both schools.

WP2 took place with two low-attaining year eight English classes in School 2. English is streamed according to academic attainment in School 2 and learners receive five hours of tuition per week. Unlike WP3 and WP4, the sample here is more restricted as it is only comprised of learners that are low attaining: the full range of attainment is not accounted for.

3.5.3 Sampling Considerations

All WPs in School 1 were comprised of 'mainstream' teaching groups. WP3 and WP4 were comprised of four Year 7 mixed-attaining and mixed-gender drama classes: two experimental groups and two comparison groups. WP1 was similar but took place with a single ICT class only. WP2 in School 2, however, differed in that two classes were low-attaining and mixed-gender Year 8 English groups. Classes were typically made up of around twenty to thirty learners.

When choosing the classes and participants within them, a nonprobability sampling strategy was used. The sample was chosen on the basis of convenience for the subject specialist teachers and the willingness of their learners to participate. This approach is described by Cohen et al. (2007) and Weathington et al. (2010) respectively:

> The selectivity which is built into a nonprobability sample derives from the researcher targeting a particular group, in the full knowledge that it does not represent the wider population; it simply represents itself. This is frequently the case in small-scale research, for example […] two or three groups of students. […] [T]his is frequently the case for some ethnographic research, action research or case study research (*ibid.*, p. 114).
>
> The most common type of nonprobability sampling is *convenience sampling.* [*italics in original*] In contrast to probability sampling, convenience sampling means that the researcher uses members of the population who are easy to find (p. 205).

A nonprobability convenience sampling method was selected because it enabled the researcher to use 'the nearest individuals' to 'serve as respondents' (Cohen and Manion, 1994, p. 88). In this case, classes were selected according to availability of the timetable. Next, classes were chosen based on the willingness of the learners within them to participate in the study and the willingness of parents / guardians to provide consent.

Although a convenience sampling strategy was used, there was a degree of researcher intervention to ensure some population diversity and some sense of generalizability. When selecting classes according to the availability of the timetable, care was taken to ensure that a range of pupil ages, genders and attainment formed part of the learner sample in WP3 and WP4. The exception to this is WP2: here the sample was less diverse as all learners were members of lower-attaining classes only.

Broadly speaking, the activities were piloted over a six-week period in each WP. For WP1 the activities linked computational thinking to several curriculum areas. All other WPs linked to the cross-curricular area of literacy.

3.6 Ethical Considerations

A research proposal was submitted to – and approved by - the University of South Wales, Caerleon School of Education Research Ethics Committee prior to undertaking the research.

The study been designed to achieve compliance with the BERA (British Ethical Research Association) guidelines for educational research (2011) and the BSA (British Sociological Association) statement of ethical practice (2002).

BERA (2011) guidelines state that:

> Researchers must take the steps necessary to ensure that all participants in the research understand the process in which they are to be engaged, including why their participation is necessary (p. 5).
>
> [R]esearchers must also seek the collaboration and approval of those who act in guardianship (e.g. parents) or as 'responsible others' (i.e. those who have responsibility for the welfare and well-being of the participants e.g. social workers) (p. 7).

Following the BERA guidelines, steps were taken to gain consent from the participants, their parents/carers and the 'responsible others' at the schools where the research took place. Each of these will now be discussed in turn. Please see *Appendix C* for copies of the forms that were used to secure consent from the different parties involved.

3.6.1 School leaders

The respective CPOs (Child Protection Officers) at both schools also served as representatives of the school's SMT (Senior Management Team). Ethical approval was sought from the CPOs at both schools in addition to the University's Research Ethics Committee. Discussions with the school leadership team included the mechanisms for securing parental and learner consent in addition to steps taken in order to secure the anonymity of learners involved. The right to publish findings was retained but with a reassurance that both the school and the participants involved would not be named.

The involvement of school leaders was vital in securing ethical integrity and failure to do so would have been detrimental to the study, as Hopkins (2002) warns:

> Actions are deeply embedded in an existing social organization and the failure to work within the general procedures of that organization may not only jeopardize the process of improvement but existing valuable work (p. 135)

3.6.2 Teachers

Collaborating teachers in WP2-4 helped to re-develop their SoL in order to meaningfully incorporate the activities. This step was taken to ensure that no learner was disadvantaged in the breadth of the curriculum that was experienced as a result of them participating in the project. As colleagues of the teacher-researcher were involved as co-researchers, observations and outcomes did not have any impact or influence regarding performance management and school-based quality assurance processes.

Collaborating teachers were also made aware of the importance of properly administering the ethics process within their schools. This was less of a concern in School 1 as the teacher-researcher was available to assist the collaborating teacher with administering this process. In School 2 this was not the case and so additional time was scheduled to ensure ethical compliance.

3.6.3 Parents/Carers

Appropriate time was set aside to ensure parent/carer consent in addition to that of the learners themselves. It was explained to parents that where there was an agreement to provide an audio recording of learner views, all audio files would be deleted as soon as they were transcribed into written form. In terms of quantitative data collection, it was explained that it is the teacher-researcher's role to look for differences and similarities in outcomes between children who complete the activities on the computer and those who do not.

Thought was also given to reassuring parents that learners would not be penalised for opting out of the research. One learner opted of WP5. This learner was not prevented from attending the extracurricular club: the learner undertook the same activities but no data collection took place with this particular learner. No learners or parents/carers withdrew consent mid-way through data collection. However, a specific statement in the ethical briefings explained that this was a possibility at any time throughout the research with no reason required.

3.6.4 Learners

Kemmis and McTaggart (1981) provide a useful framework for securing effective ethical practice when working with an action research methodology in classroom contexts. It is recommended that the researcher accept responsibility for maintaining anonymity and confidentiality. Before such an assurance was made, the researcher needed to clearly define the scope of this statement so that it did not present a conflict with School 1 and School 2's Child Protection Policy. Participants were made aware, in an accessible language, that agreements of confidentiality did not override the legal obligation of the researcher to refer information to the relevant authorities if the welfare of the child is questioned.

Further, BERA guideline sixteen (*ibid.*) states that all children and young people should be able express a view on whether or not to provide consent in a way that is 'commensurate with their age and maturity' (p. 6).

All learners were fully informed about the purpose and nature of this study verbally before they agreed to participate. This advice accounted for the level of linguistic development of the participants. It was explained to learners verbally, for instance, that the research was interested in whether 'using a computer helped them with their thinking and their English skills'. Such simplifications were necessary in order to ensure that learners understood the purpose of the study fully.

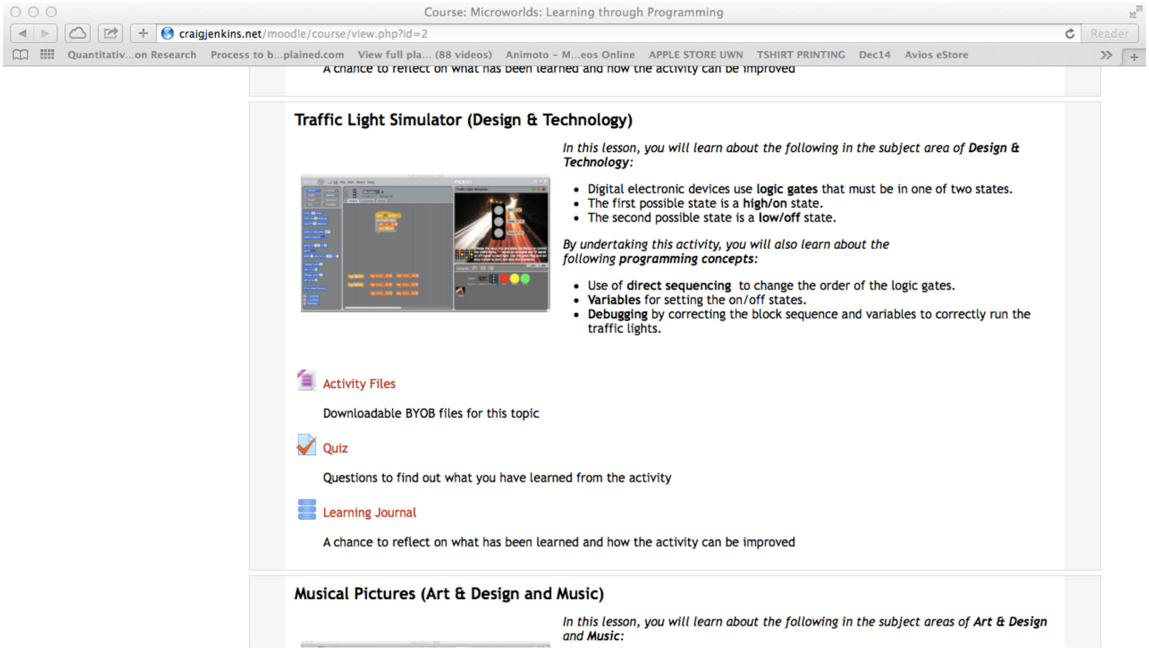
Respondent validation (*see* McCormick and James, 1983) was an important part of the study and so all participants were briefed about the findings following the completion of each work package. Participants were also informed about the impact that their involvement will have upon future practice at the school. Again this was phrased in a way that accounted for the language development of the participants and helped to secure fully-informed consent. An example of the phrasing used is: 'we found out that people did better working on the computer, so the year six's [*sic*] coming up [to secondary school] will be doing this work on the Chromebooks next year'.

4 Work Package 1: BYOB Custom Block Kits Across National Curriculum Areas

4.1 Introduction

WP1 was an exploratory pilot of a number of microworlds-based activities at School 1. The pilot was carried out in order to establish the value of using BYOB to develop microworlds and identify curriculum areas where this might be useful. for A series of microworld activities were created using BYOB for a variety of curriculum areas including mathematics, biology, history and art and design. A Moodle VLE course (see Moodle Trust, 2013a) was used to structure the microworlds-based teaching scheme. The course comprised the downloadable microworld activity files, Moodle quizzes and a series of learning journals to enable self-reflection following each activity.

The Moodle course consisted of seven topics. Each topic targeted specific computational thinking concepts and subject-specific knowledge within a curriculum area. The activities were designed with subject-specialist teachers but were piloted within discrete ICT lessons with delivery solely by the teacher-researcher.



Course: Microworlds: Learning through Programming
craigjenkins.net/moodle/course/view.php?id=2
Reader
Quantitativ...on Research
Process to b...plained.com
View full pla... (88 videos)
Animoto – M...eos Online
APPLE STORE UWN
TSHIRT PRINTING
Dec14
Avios eStore

A chance to reflect on what has been learned and how the activity can be improved

Traffic Light Simulator (Design & Technology)

*In this lesson, you will learn about the following in the subject area of **Design & Technology**:*

- Digital electronic devices use **logic gates** that must be in one of two states.
- The first possible state is a **high/on** state.
- The second possible state is a **low/off** state.

*By undertaking this activity, you will also learn about the following **programming concepts**:*

- Use of **direct sequencing** to change the order of the logic gates.
- **Variables** for setting the on/off states.
- **Debugging** by correcting the block sequence and variables to correctly run the traffic lights.

Activity Files

Downloadable BYOB files for this topic

Quiz

Questions to find out what you have learned from the activity

Learning Journal

A chance to reflect on what has been learned and how the activity can be improved

Musical Pictures (Art & Design and Music)

*In this lesson, you will learn about the following in the subject areas of **Art & Design** and **Music**:*

Figure 6: Screen Grab of one topic in the Moodle VLE Course

Figure 6 shows a screen grab of the Moodle course with a focus on the *Traffic Light Simulator* topic for the Welsh National Curriculum domain of *Design & Technology*. Learners were explicitly made aware of the objectives of each microworld activity before undertaking it. Objectives were expressed in terms of subject-specific knowledge in addition to computational thinking knowledge elements.

A description of the pedagogy of the teaching scheme is as follows. Microworld interventions were trialled in seven one-hour discrete ICT lessons (a sequence of lessons lasting a single half term). Each lesson began with a teacher-led delivery of the learning objectives that were deconstructed according to (i) the subject-specific knowledge elements (mapped to the respective NC orders) and (ii) the computational thinking concepts (*see* Brennan and Resnick, 2015) that were targeted by the activity. This was followed by a whole-class discussion of the activity and how to use the microworld. Teacher-directed questioning and learner participation was used at key points during the discussion.

After the whole-class discussion had taken place, learners were provided with around thirty-five minutes to experiment with the microworld activity. During this time, the teacher-researcher differentiated support levels in order to scaffold the activities for less able learners.

During the final twenty minutes of the lesson, learners undertook a Moodle quiz focusing on the subject-specific knowledge targeted by the microworld activity. The results of the quizzes were not stored and this data was not analysed. The quizzes were used only as a self-assessment tool for the learner to guide their reflection on the learning that had taken place.

The final five minutes of each lesson required learners to add a record to their digital learning journal. The learning journal was in fact a Moodle database activity so that by the end of the scheme a learner with full attendance would have seven separate entries for each of the activities. The learning journals provided the self-reporting data that is presented in the findings.

4.1.1 Overview of the Moodle course activities

Each topic made use of three Moodle course activities (see Moodle Trust, 2013b):

1. *Activity Files* – A Moodle **file** resource was used to enable the learner to download the BYOB microworld activity file to their 'home drives' on the school network.

2. *Quiz* – A self-marking Moodle **quiz** activity was used was used as a self-assessment tool to enable learners to self-assess their knowledge acquisition following undertaking the microworld activity. No quantitative data analysis was carried out as part of this WP, though such methods *were* incorporated into subsequent WPs.

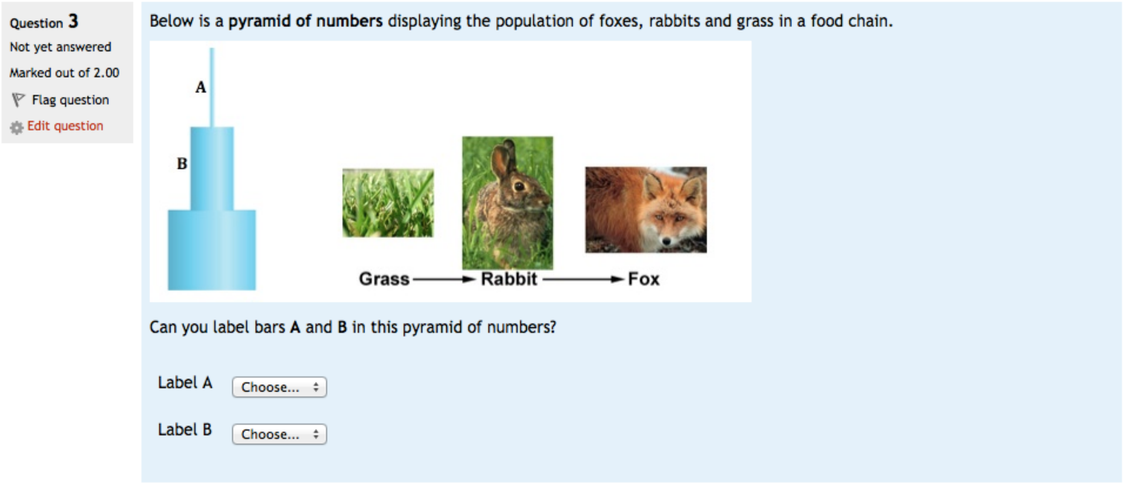
3. *Learning Journal* – A Moodle **database** enabled the self-reporting focus of the study, as described in the methods section.

4.1.2 Format of the Moodle Quiz tool

The Moodle quiz was chosen as a self-assessment tool because of the prior experience of the researcher. This project builds on an unpublished M.A research project that proposes some effective practice recommendations when use a VLE for AfL (Assessment for Learning) with a particular focus on the Moodle quiz feature (see Jenkins, 2011).

Moodle quiz assessments, like the BYOB activities, were developed in conjunction with subject-specific teachers in order to contextualize the learning locally within School 1. They were designed to enable learners to make well-informed, reflective judgments when self-reporting their knowledge acquisition following the microworld-based teaching scheme. These judgments were, in turn, expressed through the learning journals.

Quizzes consisted of five different Moodle question types: matching, numerical, multiple choice, true or false, and short answer. The quizzes were self-marking and linked to the seven curriculum areas that were explored. *Figure 7* shows an example of a matching question for the topic of pyramids of numbers in the curriculum area of Science.



Question 3

Not yet answered

Marked out of 2.00

Below is a **pyramid of numbers** displaying the population of foxes, rabbits and grass in a food chain.

Can you label bars **A** and **B** in this pyramid of numbers?

Label A: Choose...

Label B: Choose...

Figure 7: A Matching Question for the Curriculum Area of Biology

4.1.3 Design of the microworld activities

The seven microworld activities that comprised the pilot study were designed to promote some key aspects of computational thinking used in programming and Computer Science such as sequencing, branching conditions and iteration. The design of each activity, then, aimed to provide a combination of programmability using specific ideas in each domain, in order to provide an exploratory activity.

The data files for each of the activities discussed in this section can be viewed in *Appendix A* along with video-based demonstrations. Also included here is a microworlds activity pack that gives more detailed information on the subject knowledge addressed in each activity. Like the *RoboBuilder* project (Weintrop et al., 2012), each activity uses two distinct areas of the screen: one area for programming and a second area to view the outcomes of actions. Learners fit together blocks in the scripting pane in order to make changes to sprites on the stage.

This section incorporates a series of tables that decompose the subject-specific and computational thinking knowledge elements that were targeted by each microworld activity. Subject-specific knowledge elements were ratified by subject-specialist teachers and mapped to specific areas of the National Curriculum programmes of study – the subject orders - for that discipline. The computational thinking concepts and practices are mapped to Brennan and Resnick's framework (2012) for computational thinking with block-based programming projects in Scratch.

4.1.3.i Rocket Maze: A Microworld for Mathematics

Knowledge Mapping of Activity

Type of Knowledge	Deconstruction of Knowledge Elements
Subject-specific (NC Mapping)	• Angle of turn on a 2-D plane.
Computational Thinking (Brennan and Resnick, 2012 framework mapping)	**Computational Concepts** • *Sequences* – Placement of programming blocks to control the rocket. • *Loops* – Using a *repeat* block to replay common rocket movements. • *Data* – Assigning values to variables including thrust and angle of turn. **Computational Practices** • *Testing and debugging* – Correcting variable values to reach the moon. • *Abstracting and modularizing* – Use of named procedures for common movements.

Table 5: Knowledge Mapping of Rocket Maze

Table 5 shows the knowledge mapping for an activity called *Rocket Maze.* Learners used programming techniques of increasing difficulty, such as iteration and proceduralization, in order to guide a rocket to the moon on a simple 2D plane. In a form that recalls Papert's Turtle Graphics microworld, learners can use the directional block RT and define the angle of turn variable as 90. The learner will find that a rocket rotates around its centre through an angle of 90 degrees and can explore the impact of increasing and decreasing the angle of turn.

This microworld is designed for the Welsh National Curriculum area of mathematics. In particular, it addresses the KS3 requirement for learners to 'explain and use angle properties of 2-D shapes' and the KS4 requirement to 'calculate [...] angles [...] associated with common shapes' (DCELLS, 2008c, pp. 19; 32).

4.1.3.ii Traffic Light Simulator: A Microworld for Design & Technology

Table 6 shows the knowledge mapping for *Traffic Light Simulator.* Learners made use of a new *set* block in the scripting pane to control two variables: colour of light and state of light. After specifying a colour of either red, amber or green; learners control the light on/off state by specifying a binary input bit. The aim is to explore different variable values and sequences in order to simulate a set of traffic lights.

This microworld activity addresses part of the systems and control component of the design and technology curriculum area: 'learn about the properties and characteristics of electrical/electronic and mechanical components' (DCELLS, 2008a, p. 15).

Knowledge Mapping of Activity	
Type of Knowledge	**Deconstruction of Knowledge Elements**
Subject-specific (NC Mapping)	• Digital electronic devices use inputs and outputs of logic gates that must always be in one of two states. • The first possible state is a high/on state. • The second possible state a low/off state.
Computational Thinking (Brennan and Resnick, 2012 framework mapping)	**Computational Concepts** • *Sequences* – Sequencing blocks to change the order of the logic gates. • *Data* – Learners change the state of each light by specifying binary input bit values of '0' or '1'. **Computational Practices** • *Testing and debugging* – Modelling of block sequence and variable values in order for the traffic lights to function correctly.

Table 6: Knowledge Mapping of Traffic Light Simulator

4.1.3.iii Twinkle Little Star: A Microworld for Music

The *Twinkle Little Star* activity (*see Table 7*) involves learners interacting with a new *play note* block in the scripting pane. Learners are required to define two variables for note name and number of beats. When the program is run, a sound is played and the costume of the main stave sprite changes to reflect the note that is being played. The goal of the activity is to recreate the song *Twinkle Little Star* by linking through exploring different combinations note names to the respective musical notation on a treble stave.

This activity has the potential to address more than one area of the music curriculum. Firstly, the use of ICT in music applications is itself a specific requirement of the curriculum: 'realise music using ICT and music technology' (DCELLS, 2008d, p. 14). Despite having a short-term goal to construct a specific sequence of notes, this activity can also be extended to enable learners to make their own compositions. This maps to a further descriptor of the curriculum: 'compose using ICT and music technology (*ibid.*, p. 15).

Knowledge Mapping of Activity

Type of Knowledge	Deconstruction of Knowledge Elements
Subject-specific (NC Mapping)	• Notation on the treble clef. • Rhythm notation, e.g. crotchets and minims, and the number of equivalent crotchet-beats. • Intervals and the difference in pitch between notes.
Computational Thinking (Brennan and Resnick, 2012 framework mapping)	**Computational Concepts** • *Data* – Learners specify variables for note name and number of beats. • *Loops* – Use of a *repeat* block to replay sequences of notes.

Table 7: Knowledge Mapping of Twinkle Little Star

4.1.3.iv Feudal Society & Pyramids of Numbers: Microworlds for History & Science

The *Feudal Society* (*see Table 8*) and *Pyramids of Numbers* (*see Table 9*) activities share some common features in their construction so are discussed together in this sub-section.

In both activities, learners are required to explore the impact of changing four input variables in the scripting pane. For *Feudal Society* these variables represent the number of kings, barons, knights, and peasants in a feudal society structure; for *Pyramids of Numbers* these are tertiary consumers, secondary consumers, primary consumers, and producers in a food chain.

When both programs are run, the four corresponding variable monitors on the stage are updated and nested conditionals display a costume according to the values entered. The message explains how the environment is affected by the input values specified and guides the learner to refine their inputs appropriately. If a learner disproportionately decreases the number of knights in *Feudal Society*, for example, a message will be displayed explaining that the society has been invaded without adequate protection. In *Pyramids of Numbers*, a reduced population of producers will display a message explaining that there is a food shortage for primary consumers. The aim is to discover, through exploration, the optimum conditions for a sociological feudal class structure or sustainable food chain.

The *Feudal Society* microworld is mapped to the following descriptor within the *history* curriculum requirements: ‘how the coming of the Normans affected Wales and Britain between 1000 and 1500’ (DCELLS, 2008e, p. 14).

Knowledge Mapping of Activity	
Type of Knowledge	**Deconstruction of Knowledge Elements**
Subject-specific (NC Mapping)	• The ranks and power structure of medieval government. • The king as the ruler of a country and owner of all land. • The barons as suppliers of knights. • The knights as the fighters of the land. • The peasants or villeins as the workers of the land.
Computational Thinking (Brennan and Resnick, 2012 framework mapping)	**Computational Concepts** • *Data* – Learners specify variables for number of kings, barons, knights and peasants. **Computational Practices** • *Modelling* – Learners explore the impact of changing variables on the feudal society.

Table 8: Knowledge Mapping of Feudal Society

The *Pyramids of Numbers* microworld maps to the following descriptor within the *science* curriculum requirements: ‘the interdependence of organisms and their representation as food webs, pyramids of numbers and simple energy-flow diagrams’ (*ibid.*, 2008f, p. 14).

Knowledge Mapping of Activity	
Type of Knowledge	**Deconstruction of Knowledge Elements**
Subject-specific (NC Mapping)	• The producer is a green plant at the start of the food chain. • Consumers are animals at the other stages of the food chain. • The nomenclature of primary, secondary and tertiary consumers. • Pyramids of numbers for showing population numbers in a food chain.
Computational Thinking (Brennan and Resnick, 2012 framework mapping)	**Computational Concepts** • *Data* – Learners specify variables for number of tertiary consumers, secondary consumers, primary consumers and producers. **Computational Practices** • *Testing and debugging* – Learners explore the impact of changing variables on the virtual forest.

Table 9: Knowledge Mapping of Pyramid of Numbers

4.1.3.v Sentence Generator: A Microworld for English

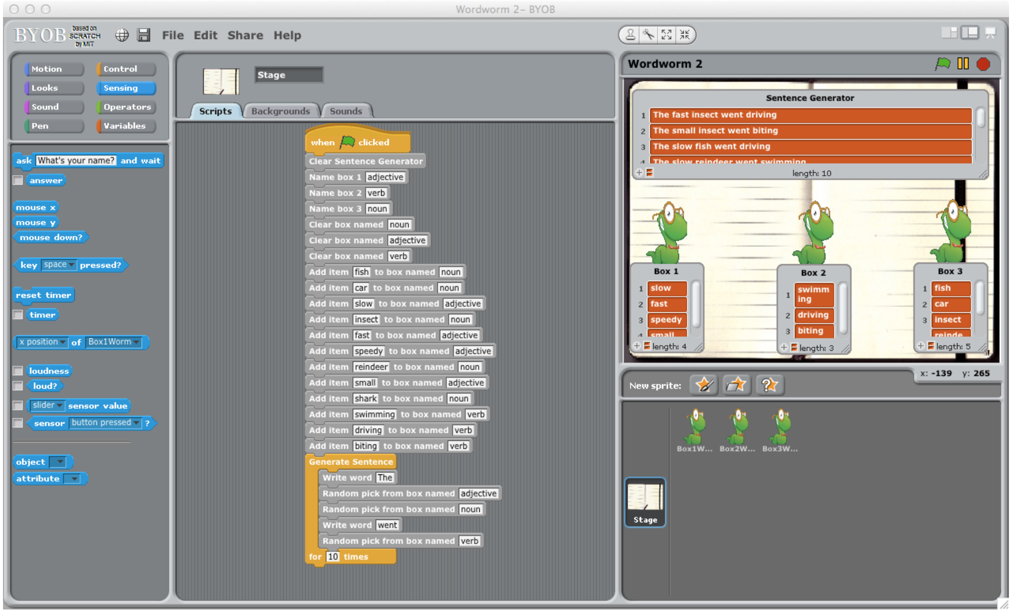
The *Sentence Generator Activity* (*see Table 10*) is directly influenced by the work of Sharples (1985) and also Goldenberg and Feurzeig (1987) discussed previously. A screen grab of this activity accompanies the knowledge mapping in *Figure 8.* The activity can be understood in terms of four steps. It is important to reiterate that *Appendix A* provides a dynamic video demonstration in addition to the static text description provided here.

Step 1 - Adding labels to the three boxes

There are three boxes of words at the bottom of the stage. By using the block "name box" the learner decides what class of word that particular box will represent. In the screen grab, the learner has decided that the first box will be called "adjective", the second box will be called "verb" and the third box will be called "noun".

Step 2 – Putting words in the different boxes

The "add item" block is where learners place examples of different words into the three boxes at the bottom of the stage. You can see from the screen grab that the learner has placed the words "fish" and "car" in the box she has called "noun". She has placed the word "slow" into the box she has called "adjective".



```
when clicked
Clear Sentence Generator
Name box 1 adjective
Name box 2 verb
Name box 3 noun
Clear box named noun
Clear box named adjective
Clear box named verb
Add item fish to box named noun
Add item car to box named noun
Add item slow to box named adjective
Add item insect to box named noun
Add item fast to box named adjective
Add item speedy to box named adjective
Add item reindeer to box named noun
Add item small to box named adjective
Add item shark to box named noun
Add item swimming to box named verb
Add item driving to box named verb
Add item biting to box named verb
Generate Sentence
    Write word The
    Random pick from box named adjective
    Random pick from box named noun
    Write word went
    Random pick from box named verb
for 10 times
```

Sentence Generator
1 The fast insect went driving
2 The small insect went biting
3 The slow fish went driving
4 The slow reindeer went swimming
length: 10

Box 1
1 slow
2 fast
3 speedy
4 small
length: 4

Box 2
1 swimm ing
2 driving
3 biting
length: 3

Box 3
1 fish
2 car
3 insect
4 reinde
length: 5

Figure 8: Screen Grab of Sentence Generator

Step 3 – Writing the sentence structure

Everything placed in the "generate structure" C-shaped block will appear as a sentence in the sentence generator at the top of the stage. This can be repeated using the "for times" part of the block. If a "write word" block is used, this word will stay exactly the same each time a new sentence is generated. Using the "random pick from box named" block will substitute whatever box name is typed in with a random word taken from that box.

Step 4 – Generate sentences and substitute words

When learners click the green flag icon, a number of sentences are generated according to the sentence structures that have been specified in step 3. Where the "random pick" block has been used, these words will be substituted at random with a pick of one word taken from that box.

Knowledge Mapping of Activity

Type of Knowledge	Deconstruction of Knowledge Elements
Subject-specific (NC Mapping)	• Adjectives give more detail about a noun. • A noun names things. • A verb explains something that is happening. • Creating a simple sentence consisting of a clause with a subject and a verb.
Computational Thinking (Brennan and Resnick, 2012 framework mapping)	**Computational Concepts** • *Sequences* – Correct placement of blocks in order to generate random sentences. • *Loops* – Use of a *generate sentence for times* block to specify the number of sentences to produce. • *Data* – Assigning values to box names and placing items within them. **Computational Practices** • *Testing and debugging* – Adjusting the construction of the sentence if sentences that are generated do not make grammatical sense. • *Abstracting and modularizing* – The *generate sentence* procedure generates new sentences.

Table 10: Knowledge Mapping of Sentence Generator

The *Sentence Generator* activity maps to multiple descriptors of the curriculum for English. The first part of the activity, where learners name different grammar components and give examples of them, enables an exploration of 'standard forms of English: nouns, pronouns, adjectives, adverbs, prepositions, connectives and verb tenses' (DCELLS, 2008g, p. 18). The second part of the activity, where the learner specifies different sentence structures, enables them to investigate 'the range of sentence structures effectively to enhance the fluency and coherence of their writing' (*ibid.*).

4.1.3.vi Musical Pictures: A Microworld for Art and Design & Music

The *Musical Pictures* activity (*see Table 11*) makes use of the Picoboard sensor board hardware that was described in the literature review. The microworld enables learners to experiment with sensor board inputs in order to develop their own creative artistic and musical creations.

Five new blocks in the scripting pane facilitate exploration with the stage visuals and sound: *make note*, *make volume*, *make paint size*, *make paint colour* and *make vertical position*. These variables can be controlled using the Picoboard hardware by way of three reporter blocks called *amount of sound*, *amount of light* and *position of slider*. Four standard scratch mathematical operator blocks are used for *addition*, *subtraction*, *multiplication* and *division* of the sensor board values. These can be stacked within the five motion and sound blocks whilst the sensor board reporter blocks can in turn be stacked within the mathematical operator blocks. Learners control a paintbrush on the stage and sound from the speakers by experimenting with changing values in the operator blocks and by changing environmental conditions detected by the sensor board hardware. This means that the learner is able to use the microworld to create their own visual artworks and musical compositions.

This activity simultaneously addresses requirements within two areas of the curriculum. Learners are able to make their own musical creations using the digital technology of the computer-based microworld. This maps to the following descriptor in the music curriculum: 'compose using ICT and music technology' (DCELLS, 2008d, p. 14). Learners are also able to create their own visual compositions by controlling the paintbrush on the stage. This maps to the following descriptor in the curriculum for art and design: 'pupils should design and make both imaginatively and expressively [...] they should use a variety of materials, e.g. [...] digital-based media [and] processes, e.g. [...] digital-based processes' (DCELLS, 2008h, p. 15).

Knowledge Mapping of Activity

Type of Knowledge	Deconstruction of Knowledge Elements
Subject-specific (NC Mapping)	• Composition of music using ICT and music technology. • Artistic design using digital-based media and processes.
Computational Thinking (Brennan and Resnick, 2012 framework mapping)	**Computational Concepts** • *Sequencing* – Learners hierarchically place sensor reporter blocks within customized blocks that control the paintbrush and sound. • *Data* – Assigning values to variables to control the paintbrush, such as *make paint colour*; and sound, such as *make volume*. Learners work with hardware sensors that are used in real-world systems. • *Operators* – Learners use the four basic mathematical operators in their programming constructions.

Table 11: Knowledge Mapping of Musical Pictures

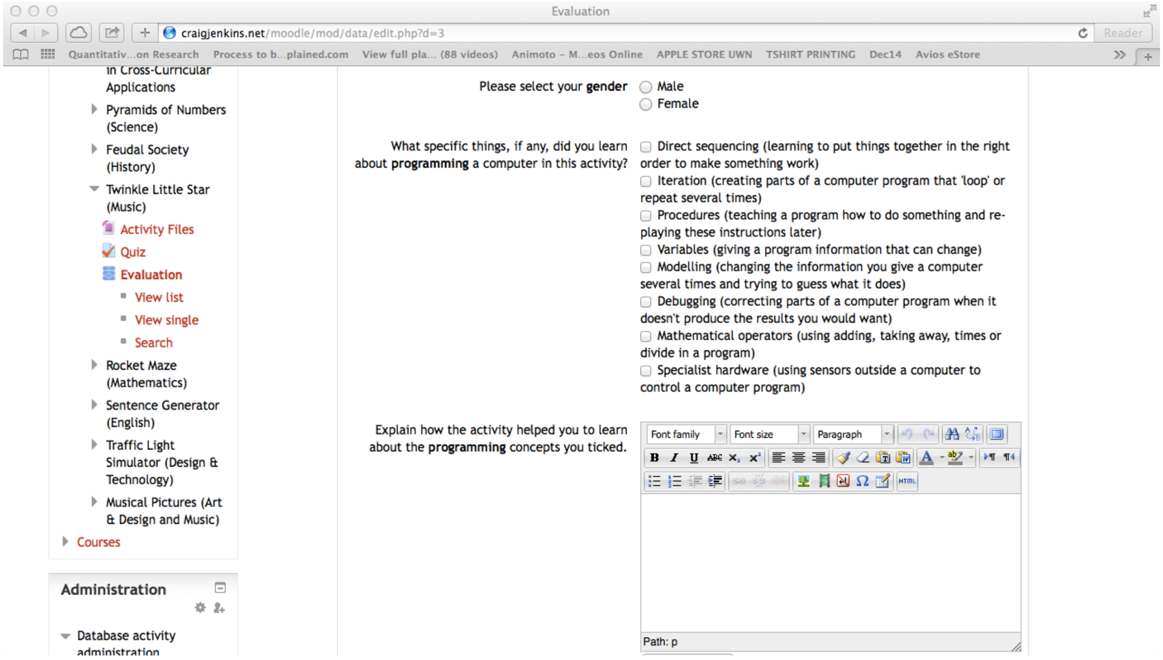
4.2 Methods

The self-report study was conducted with a group of 29 mixed-attaining and mixed-gender (15 girls; 14 boys) learners aged eleven and twelve in Year 7. Over a period of one half term (around seven weeks), the activities were undertaken by the learners within their standard timetabled ICT lessons (one hour per week).

Exploratory data were gathered from two sources: unstructured active-participant observation as a teacher-researcher and a short online survey, in the form of a learning journal. Learners were asked to update their learning journals at the end of each session, resulting in seven entries over the course of the half term. They were required to identify improvements in their subject-specific and computational thinking knowledge whilst also evaluating the effectiveness of the BYOB microworld activity.

4.2.1 Format of the Moodle Database Learning Journals

As with the Moodle quiz, a Moodle database was chosen for the structured learning journals as this builds on the prior experience of the researcher. *Figure 9* shows a screen grab of a portion of the Moodle database activity. The learning journal that was maintained was updated as a plenary activity at the end of each topic in the microworlds-based teaching scheme. The learning journal was used to provide the self-reporting data that is analysed in the study findings.



Evaluation
craigjenkins.net/moodle/mod/data/edit.php?d=3
Reader
Quantitativ...on Research
Process to b...plained.com
View full pla... (88 videos)
Animoto – M...eos Online
APPLE STORE UWN
TSHIRT PRINTING
Dec14
Avios eStore
in Cross-Curricular Applications
Pyramids of Numbers (Science)
Feudal Society (History)
Twinkle Little Star (Music)
Activity Files
Quiz
Evaluation
View list
View single
Search
Rocket Maze (Mathematics)
Sentence Generator (English)
Traffic Light Simulator (Design & Technology)
Musical Pictures (Art & Design and Music)
Courses
Administration
Database activity administration
Font family
Font size
Paragraph
HTML

Please select your **gender**
○ Male
○ Female

What specific things, if any, did you learn about **programming** a computer in this activity?
□ Direct sequencing (learning to put things together in the right order to make something work)
□ Iteration (creating parts of a computer program that 'loop' or repeat several times)
□ Procedures (teaching a program how to do something and re-playing these instructions later)
□ Variables (giving a program information that can change)
□ Modelling (changing the information you give a computer several times and trying to guess what it does)
□ Debugging (correcting parts of a computer program when it doesn't produce the results you would want)
□ Mathematical operators (using adding, taking away, times or divide in a program)
□ Specialist hardware (using sensors outside a computer to control a computer program)

Explain how the activity helped you to learn about the **programming** concepts you ticked.

Path: p

Figure 9: Learning Journals using a Moodle Database

Two fields made use of a radio buttons type, namely *gender* and *year group*, for categorization purposes. These fields allowed only one choice to be made. A presence check ensures that one choice *must* be selected before an entry can be submitted.

A check box field type was used for learners to identify the aspects of computational thinking and subject-specific knowledge that they felt had improved following the BYOB activity. Learners were presented with a series of check boxes and were required to select multiple (if any) areas where they identified an improvement.

Text area field types behave in a similar way to text boxes but can span multiple lines and therefore encourage a greater richness of response. Text area fields were used as probing mechanisms in order for learners to justify the check boxes that were selected. This field type was also used so that learners could highlight strengths and improvement points when evaluating the BYOB activities.

4.3 Results and Discussion

The use of the block-based Scratch platform may seem like a curious choice for undertaking programming activities in Year 7. Some secondary teachers may suggest that Scratch is more suited to primary school levels where it is assumed that learners' first programming experiences are gained. For some, therefore, Scratch is too elementary for use in Key Stage 3 teaching. A baseline self-assessment question in the learning journal showed that this assumption of prior experience in the primary school was not strongly supported by the experience of the students. A majority of 66% said that they had never encountered programming activities during their primary school education. Perhaps secondary level educators in ICT and computing need to be mindful of this knowledge gap when incorporating computational thinking and programming learning experiences into their classrooms.

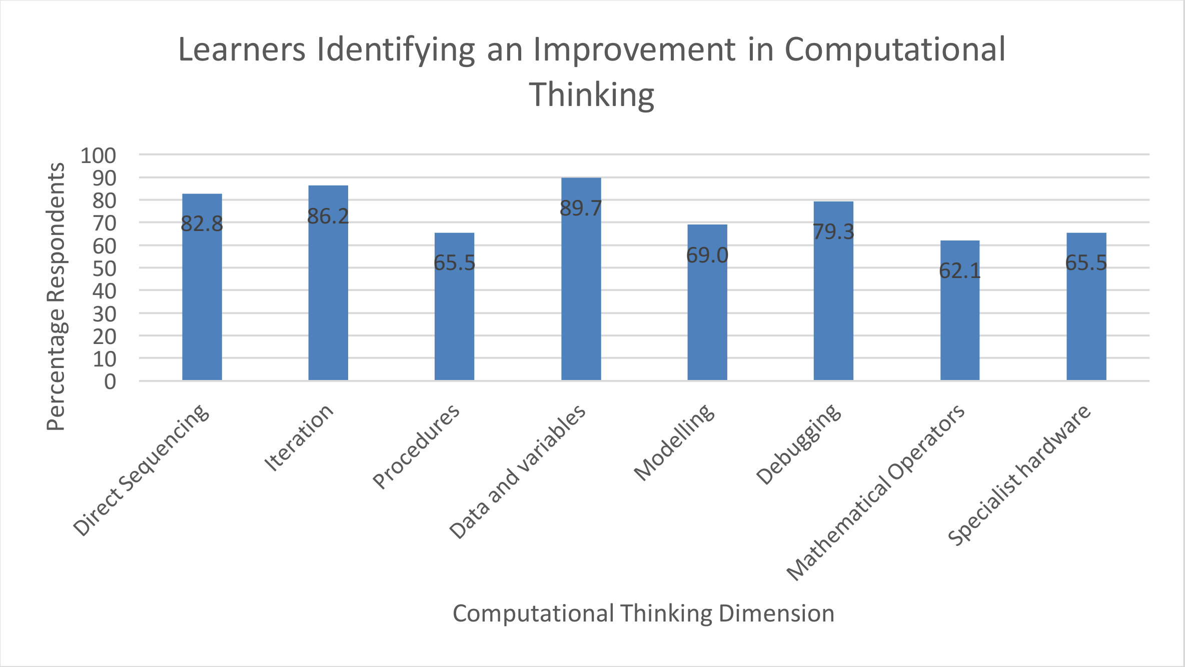


Figure 10: Percentage of learners identifying improvements in computational thinking dimensions

Following the microworld activities, nearly all learners (97%) indicated that their computational thinking had improved. Only one learner (accounting for the remaining 3%) felt that the activities did not bring about an improvement at all. Learners were then asked to identify the specific programming concepts that they knew more about following the pilot activities (*see Figure 10*). Most learners felt that their knowledge about sequencing, looping and data and variables had improved considerably.

The relatively high response that learners gave to sequencing and data and variables is expected as these appear as targeted computational thinking dimensions in the design of most of the microworld activities (*see* section 4.1.3). More surprising was the high score given to looping because this concept was only targeted in the two microworlds activities for English and mathematics. This indicates that learners were able to relate *bounded* iteration, where a set of instructions is repeated for a set number of times, to where this was used in the microworlds teaching scheme. The high response given to looping mirrors the high responses given to the subject-specific knowledge aspects in these particular activities.

Knowledge Aspect	Curriculum Area	%	n
Use of angles	Mathematics	93.1%	27
Digital electronic devices and the use of logic gates	Design Technology	51.7%	15
High/on and low/off states in logic gates	Design Technology	96.6%	28
Musical notes, their names and pitches on the treble clef	Music	34.5%	10
The difference between crotchet and minim rhythms	Music	31.0%	9
Creating simple compositions using a computer	Music	44.8%	13
The power structure of medieval government and the Feudal Society	History	79.3%	23
The relationship between kings, barons, knights and peasants	History	72.4%	21
Pyramid of number diagrams for showing population numbers in a food chain	Biology	86.2%	25
The difference between producers and consumers	Biology	86.2%	25
The difference between adjectives, nouns and verbs	English	86.2%	25
Understanding simple sentence patterns	English	75.9%	22
Creating simple artworks using a computer	Art	13.8%	4

Table 12: Percentage of learners identifying an improvement in subject-specific knowledge

A smaller majority of learners thought that they understood more about mathematical operators after undertaking the activities. Mathematical operators were only used in one activity for the curriculum areas of art and design and music. It is interesting to note that the subject-specific knowledge for this microworld also returned less positive results. The lower score that was given to modelling and proceduralization may indicate a lack of understanding about how these specific aspects of computational thinking related to the activities that were undertaken. These two findings have not been omitted, though the lack of understanding demonstrated by learners here is a limitation. During the whole-class discussions that took place each lesson, the meaning of these dimensions of computational thinking were explained to learners. During the learning journal phase of the lessons, teacher intervention in the form differentiated support was used to explain again what these terms mean. Nevertheless, the data shows that learners still appeared to struggle with identifying where these aspects of computational thinking were used.

Learners were also asked to identify improvements in their knowledge in the curriculum areas that had been brought about by undertaking the activities (*see* table 12).

The 'Rocket Maze' activity, which perhaps is most closely influenced by Turtle Graphics, saw a particularly high proportion of learners identify a development in the area of mathematics. Learner A makes a particularly strong link between the mathematical concept of angles on a 2D plane and the computational concept of modelling:

> We did rocket maze and we had to put numbers into the boxes to make the rocket get to the moon. The computer programe was programmed for each different move e.g. turn left 90 degrees. We had to use our maths skills as well as we had to write in the number of degrees we wanted it to go forward. Also how much power we needed it to go forward. We had to keep trying for the forward as we did not know how powerful it will be [*sic*] (*Learner A*).

The four knowledge aspects that related to food chains in biology and the feudal system in history were also identified by a large majority of learners. Something that is shared by the biology, history and mathematics microworlds is the emphasis that these activities give to modelling and debugging the data that is inputted. In the mathematics microworld this involved changing the angle of the rocket, in the history microworld changing the number of peasants and in the biology microworld changing the number of producers.

Of the microworlds in the apparent creative subject areas of art and design, music and English there was much greater variation. Only a minority of learners felt that they had improved in the former two curriculum areas. The 'Musical Pictures' activity, for example, required learners to change inputs and experiment with placing operators in order to control the on-screen image and sound.

This poor result may be attributed to a lack of teacher input. The focus on discovery learning may require pedagogical refinement to incorporate opportunities for further teacher instruction. An adjustment to the pedagogy could include, for example, specific creative briefs and resource packs demonstrating the microworld primitives. Rieber (1992) is useful to recall here and in particular his formulation of a microworld as a bridge between instructionist and constructionist learning with computers.

The self-reporting data showed that the 'Sentence Generator' activity designed for the curriculum area of English was the most well-received by learners out of the apparent creative subject areas with 86% identifying an improvement in word classes and 76% identifying an improvement in sentence structures. The pedagogy of this microworld required additional teacher scaffolding in the form of dividing the activity into four distinct steps (*see* section 4.1.3.v). The additional guidance given here may explain why learners felt they improved more with this particular activity. The importance of teacher scaffolding when learning programming is supported elsewhere in, for example, Simmons et al. (2003).

Further, some respondents felt that the microworld activities could be improved by paying more careful attention to the design of the microworld and its teaching scheme. In particular, it was suggested that steps should be taken in order to maximize learner engagement with the activities:

> Make them more interesting. Bright colours and cartoons. Better cartoons. Please don't take any offence about your present cartoons, it's just that they are a bit boring [*sic*] (*Learner B*).

The portion of the response 'please don't take any offence' raises an important issue. Learners were explicitly instructed to provide honest and critical feedback in the evaluation. Additionally, learner voice groups convene regularly on both a departmental and school-wide level so learners should be comfortable with voicing their opinion. Nevertheless, the impact of differences in the exercise of authority between students and teachers remains as a potential limitation to the findings.

Though the learner feedback focuses here *intrinsic* engagement in the microworld itself, which is to a certain extent constrained by the software package being used, there are many other *extrinsic* ways to raise engagement in the development of the accompanying teaching scheme. In WP4, for instance, workbook resources and visual presentation aids were produced to accompany the microworld activities (*see* Appendix F). This provided learners with a child-friendly explanation of the subject-specific concepts needed to access the microworld. WP5 developed the microworlds teaching scheme even further by incorporating a competitive 'challenge' task brief in order to raise engagement (*see* Appendix G).

4.4 Conclusions and Limitations

The findings derived from the pilot study were useful in helping to form the guiding questions (GQ) that linked together the subsequent WPs.

The pilot established, albeit tentatively since the data relied entirely on the mechanism of self-reporting, that the microworld-based learning approach had some positive impact in terms of acquiring computational thinking concepts. This is an encouraging feature of the pilot that suggests the important part that microworlds may play in the classroom of the future, particularly in the context of the renewed emphasis on programming as a core activity. This helped form GQ1 and a focus on the potential of a microworld-based approach for developing aspects of computational thinking.

More variability, however, was seen in the data focusing on subject-specific knowledge within curriculum areas. The highest scoring activity here was the *Rocket Maze* activity that focused on mathematics. In the creative subject areas of art and design and music, however, learners self-reported improvements of only 38% and 28% respectively. The exception to this was the *Sentence Generator* activity focusing on English, where the majority of learners self-reported an improvement.

The focus on literacy development for GQ2, however, was not solely a result of the positive self-reporting data for this curriculum area. Literacy development figures as a priority both nationally (*see* Welsh Government, 2011) and institutionally at the school where the researcher is employed as a teacher (*see* Estyn, 2016 for a recent example). Additionally, there was previous research in this curriculum area (Sharples, 1985; Goldenberg and Feurzig, 1987) and it was thought that a revival of these ideas using block-based programming technologies could be an interesting focus for microworlds-based research activity.

Finally, the learning journal data collected here saw learners making suggestions about the design and teaching scheme that would be useful for other educators. An important part of the dissemination of this project is to offer effective practice recommendations for other educators wishing to take microworld-based learning into their own classrooms. GQ3, therefore, focuses on establishing such a set of recommendations.

.

5 Work Package 2: The Poem Generator Custom Block Kit in English Lessons

5.1 Introduction

When identifying creative curriculum areas that lend themselves to learning about computational thinking, the previous microworld activities designed for areas such as art and design and music appeared to be apparently less susceptible to a microworlds-based pedagogy in comparison to curriculum areas such as English. This may indicate a flaw in the design of these activities as opposed to an inherent unsuitability of the microworlds-based approach with these disciplines. Nevertheless, the self-reporting data gathered at School 1 led to a narrower focus in School 2 that was targeted towards specific aspects of cross-curricular **literacy** development.

Building on the work of Sharples (1985) and Goldenberg and Feurzig (1987), the previous microworld activity that focused on the curriculum area of English was built using the off-shoot of Scratch called BYOB. Here a refined activity was created using the online iteration of BYOB known as Snap!.

5.1.1 Design of the Custom Snap! Block-Kit

To facilitate the ‘low floor’ (see Rieber, 2004) and ease-of-access that are pedagogic characteristics of the microworlds-based learning approach, learners were provided with two versions of the microworld. The first provided a working exemplar of the microworld pre-populated with sample sentence structures and words. Learners were then given blocks in a ‘blank’ microworld to encourage experimentation.

The creators’ original rationale for Snap! and BYOB was to retain the standard blocks of Scratch for its younger audience whilst enabling a set of additional custom-built blocks to be created (*see* Mönig and Harvey, 2009). Snap! was used here to create a new, simplified set of blocks that ‘lowers the floor’ of programming to generate poetry. It achieves this by concealing the complexity of the microworld in order to increase accessibility for first-time programmers.

As with the BYOB activity for English created in WP1 (*see* section 4.3.1.v), the Snap! microworld can be expressed in four steps. Six custom blocks were created to enable learners to progress through these steps (*see* Table 13), which are now discussed in turn. Please *see Appendix D* on the data disc for the microworld XML data file to launch this activity in Snap!. Also contained here is a screen capture video of the activity in use.

Step	Description	Custom blocks used
1	Adding labels to the boxes	• Add label
2	Putting words in the different boxes	• Add words
3	Writing the sentence structures	• New Line • Write • Pick
4	Generating sentences and substituting words	• Generate

Table 13: Description of the custom block kit in Snap!

5.1.1.i Step 1 - Adding labels to the three boxes

Box labels were used to describe word class of all words held within that box, e.g. adjective, noun, verb. The **add label** block assigns a box label to each of the box numbers to make them easier to work with. There are two inputs requiring the learner to specify the box number and a text label for each box. This makes it easier to add words to particular boxes since learners no longer need to remember the corresponding box number after it has been assigned a particular text label initially.

5.1.1.ii Step 2 – Putting words in the different boxes

The **add words** block enables words like 'dog' or 'cat' to be held in a box named 'noun' whereas 'run' and 'jump' are placed in a box named 'verb'. An expandable text input requires the learner to add a sequence of words and a second text input requires that they specify the box that they would like to add the words to. The pick block then uses the three word boxes (lists) for the random picks that are made when generating poems.

5.1.1.iii Step 3 – Writing the sentence structures

A **new line** block performs the function of combining word blocks placed within the C-shape before displaying them in a single new line. A *status reporter* variable on the stage provides learners with a step-by-step explanation of what operations the block is carrying out at a given time. The block first deletes all data currently contained within a list called *picker*. The picker list is used to keep track of the text contained in the current line. The block then looks at other blocks contained within its command C shape. It concatenates the text within these blocks before adding to a further list called *Poem Generator*. A list monitor on the stage displays the resulting output.

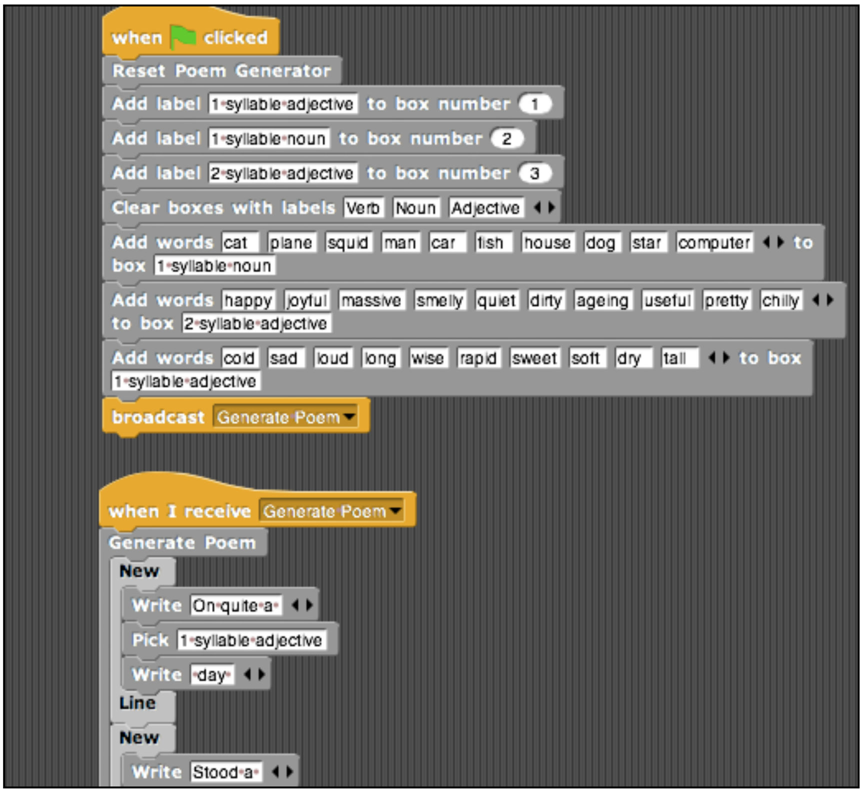
The **write** block is used to write the same word each time a new line is generated, for example always including 'the' at the beginning of the line. This type of block is placed within the C-shape of the *new line* block. It has an expandable text input that concatenates the text entered and adds it to the *picker* list in order to continue assembling the current line. Note that text contained in the write block is constant and does not require any randomization as per the *pick* block.

The **pick** block is used to identify one of the boxes and select a random word that is contained within it. If the word 'noun' is entered, for example, this could be substituted with 'dog' or 'cat' in the sentence. The block looks at the label that is specified in the text input and matches it with the labels of the three boxes or lists containing the different word types. It then picks a word from the desired box, or item from the list, at random. This random pick is in turn added to the current line that is managed by the picker list.

5.1.1.iv Step 4 - Generating sentences and substituting words

The **generate poem** block is used to repeat the lines inside the c-shape for a set number of times, making random substitutions each time a pick block has been used. It has two inputs: a command C-shape and number. Any blocks contained within the c-shape are looped for the number of times specified. This makes it easy to generate several poems with random picks from the boxes each time.

5.1.1.v Example of the microworld



```
when [flag] clicked
Reset Poem Generator
Add label [1 syllable adjective] to box number (1)
Add label [1 syllable noun] to box number (2)
Add label [2 syllable adjective] to box number (3)
Clear boxes with labels [Verb] [Noun] [Adjective] ◂▸
Add words [cat] [plane] [squid] [man] [car] [fish] [house] [dog] [star] [computer] ◂▸ to
box [1 syllable noun]
Add words [happy] [joyful] [massive] [smelly] [quiet] [dirty] [ageing] [useful] [pretty] [chilly] ◂▸
to box [2 syllable adjective]
Add words [cold] [sad] [loud] [long] [wise] [rapid] [sweet] [soft] [dry] [tall] ◂▸ to box
[1 syllable adjective]
broadcast [Generate Poem ▾]

when I receive [Generate Poem ▾]
Generate Poem
  New
    Write [On quite a ] ◂▸
    Pick [1 syllable adjective]
    Write [ day ] ◂▸
  Line
  New
    Write [Stood a ] ◂▸
```

Figure 11: The scripts area of the Poem Generator in Snap!

An example of how learners may interact with the microworld is shown In the scripts area (*see* Figure 11). Learners must first (i) decide what word class each box will represent and (ii) populate these boxes with examples of that type of word. Learners achieve this by slotting together a series of pre-built programming blocks with space to add their own text. Learners adopt a similar process to specify the word and line structure for the type of the poem they would like to generate. First, learners add blocks to specify where they would like constant words and random picks from the boxes to appear. This is then used to generate random poems when the program is run.

When they run the poem generator by clicking the green flag button, the outcomes are shown on the stage area (*as in* Figure 12; please *see* also appendix D). The stage is divided into five panels. The three smaller panels at the top represent the three lists of different types of words. The tall box running down the right-hand side of the screen shows the progress of assembling the current line. The large panel in the bottom-left is where the poems are displayed. The program generates lines and entire poems depending on the sentence structures specified using the *write* and *pick* blocks. Learners, in turn, are able to discover the rules governing written English.

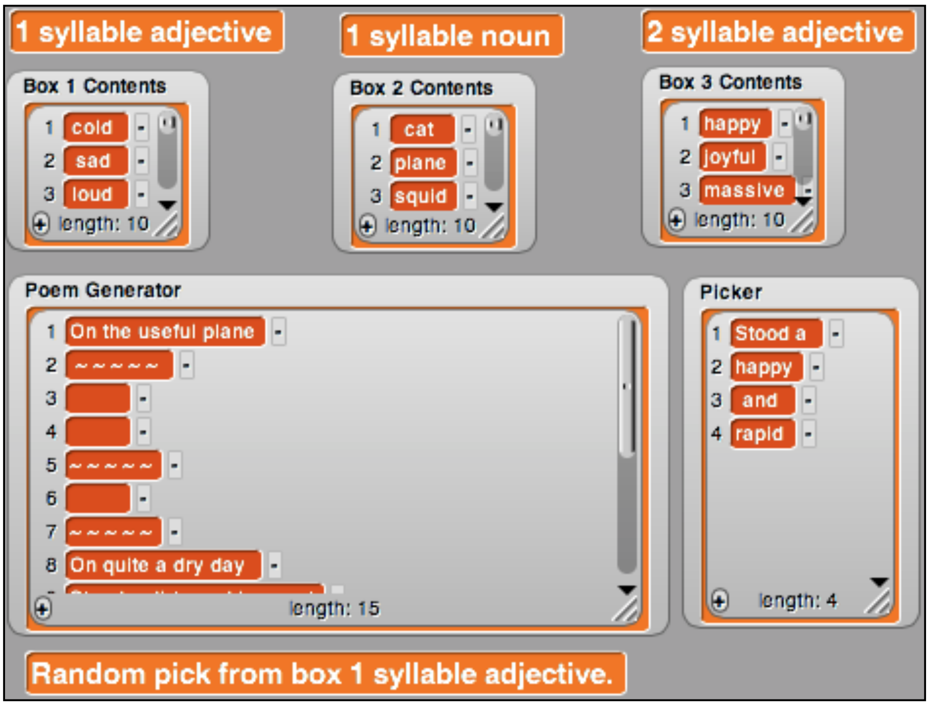


Figure 12: The stage area of Poem Generator in Snap!

5.2 Methods

To investigate the research questions, a quasi-experimental nonequivalent comparison group design (*see* Cohen, Manion and Morrison, 2011) was used with pre-tests and post-tests for computational thinking and aspects of literacy. The test can be deconstructed into specific questions assessing three specific aspects of literacy: adapting writing structures, sentence structures, and word families. A more detailed mapping to particular expectation statements within the national Literacy and Numeracy Framework can be seen in table 14. The pre-tests and post-tests were designed by the author in conjunction with an English-specialist teacher. The difference between pre-test and post-test outcomes provided a means by which to measure any difference in computational thinking and literacy following the microworld intervention.

A possible concern with quantitative research of this kind is whether the tests are designed by the teacher-researcher to test skills only taught through the intervention, thereby reducing the value of the findings. The aspects of literacy and computational thinking taught during lessons for the intervention and comparison group, however, were identical and carefully mapped to recognised frameworks (*see* tables 14 and 15). The differences between the groups were the activities undertaken. Whereas the comparison group planned their poems using mind maps and developed them using pen-and-paper, the intervention group developed their poems by exploring within the microworld.

The study took place over the course of one half term at a secondary school in South Wales; the implementation of the microworld intervention was as follows. At the beginning of the half term, both the experimental and comparison groups were issued with identical paper-based computational thinking and literacy assessments that were co-authored by the author and the English subject-specialist teacher. Identical assessments were re-issued to both groups at the end of the half term. Around 20 minutes at the beginning of the first session and a further 20 minutes at the end of the final session was dedicated to completing the pre-tests and the post-tests.

Both groups followed the scheme of learning (SoL) unit for 'Year 8 poetry' that was devised by the school. The SoL unit devised by the school aimed to teach about different word classes and their use in common poetry forms.

Both the experimental and comparison groups were provided with different examples of fixed form poetry. The experimental microworld intervention was carried out with one of the groups and took place over three one-hour lessons in a bookable ICT suite during normal timetabled English lessons. The English teacher used whole-class discussion and incidental questioning to guide the exploration with the microworld. This was in four parts that broadly mirrored the steps shown is section 5.1.1, though a greater emphasis was given to steps 3 and 4 where learners created their sentence templates and generated the poetry. The experimental group learners populated the word lists with different examples of each word class, produced the programming code required to write a haiku and then generated their finished haiku poems. The intervention was led by the English teacher after receiving training from the author. The author was available throughout in a technical support capacity.

The comparison group, over the same number of lessons, were first required to study different poetry forms and 'spot the patterns' by looking how they were structured. Class discussion was used to establish the syllabic structure of the different fixed forms and the word classes contained within them. They were then asked to write their own poems according to the structures that were observed. They produced planning for their poems using mind maps supplemented by discussion with their peers before moving on to develop their own fixed form poems using pen and paper. In total, over the three lessons they produced poetry for three poetry forms: haiku, tanka and limerick.

The scheme of work unit designed by the school links to the following study opportunities in the KS3 programme of study for English in the National Curriculum for Wales (DCELLS, 2008):

- Use the standard forms of English: nouns, pronouns, adjectives, adverbs, prepositions, connectives and verb tenses.
- Experiencing and responding to a wide range of texts that include [...] traditional and contemporary poetry'.

5.2.1 Design of the Pre-tests and Post-tests

At the beginning of the half term, both the experimental (n = 13) and comparison (n = 14) groups were issued with five identical paper-based computational thinking assessments and a further five literacy assessments. These assessments can be seen in *Appendix D*. In order to avoid the problem of pre-test/post-test equivalency, an identical set of ten tests were then re-issued at the end of the half term.

Gain (difference between pre-test and post-test scores) was measured in lieu of post-test performance. The data analyses were carried out using IBM SPSS. Initial univariate analyses were followed by independent t-tests were used alongside Cohen's D to compare the performance of the groups.

Both the microworld activity and assessments were mapped to the following expectation statements for cross-curricular literacy development in the statutory National Literacy and Numeracy Framework (DfES, 2013):

- Adapt structures in writing for different contexts.
- Use a wide range of sentence structures choosing connectives.
- Use knowledge of word roots and families; grammar, sentence and whole-text structure.

Table 14 is an assessment matrix that maps the individual exercises in literacy undertaken by both groups to the National Literacy and Numeracy framework expectation statements.

	Exercise 1 Syllables	Exercise 2 Stress	Exercise 3 Word Types	Exercise 4 Poetry Forms	Exercise 5 Poetry Analysis
Adapt structures in writing for different contexts		X		X	X
Use a wide range of sentence structures choosing connectives	X				X
Use knowledge of word roots and families; grammar, sentence and whole-text structure.			X		X

Table 14: Assessment Matrix of Literacy Expectation Statements

Resnick and Brennan (2012) provide a framework for assessing the development of computational thinking that takes place when programming with Scratch. Specifically, four computational thinking concepts and practices were tested for: testing and debugging, abstracting and modularizing, sequencing and conditionals.

	Exercise 1 Abstraction	Exercise 2 Sequencing	Exercise 3 Debugging	Exercise 4 Conditionals	Exercise 5 Programming Concepts
Sequencing *(concept)*		X			X
Conditionals *(concept)*				X	X
Testing and debugging *(practice)*			X		
Abstracting and modularizing *(practice)*	X				X

Table 15: Assessment Matrix of Computational Thinking Concepts and Practices

Table 15 is an assessment matrix that maps the individual exercises in computational thinking undertaken by both groups to the computational thinking dimensions.

5.2.2 Participants

The sample consisted of an experimental group made up of 69% boys (n = 9) and 31% girls (n = 4) and a comparison group made up of 57% boys (n = 8) and 43% girls (n = 6). Classes were selected in an attempt to control for external factors as much as possible, whilst also maintaining a natural school setting and class dynamic.

To control for external factors, the two English classes were: (i) from the same cohort – Year 8; (ii) following the same scheme of work - poetry; (iii) taught by the same English teacher; (iv) of a similar low-attaining academic set; (v) made up of learners from comparable ethnic and socioeconomic backgrounds in the same school context.

It is important to note, however, two limitations resulting from variables that were not controllable.

First, the well-documented Hawthorne Effect[1] could have resulted in an atypical higher level of performance since the English teacher leading the intervention had a pre-existing interest in technology-enhanced learning that led to their participation in the study.

Second, the experimental and comparison groups were comprised of two low-attaining English sets. The English sets were made up of students that also belonged to a mix of discrete ICT sets. A variability in ICT teaching and exposure to the Scratch programming environment, along with the different ICT capability of learners, delimits the findings of the study since sampling only accounted for English attainment.

Additionally, generalizability of the findings are restricted by the context of the school in which the study was undertaken. The most suitable Estyn (2010) inspection report at the time the intervention took place reveals an educational provider where free school meal entitlement (serving as an indicator of socioeconomic disadvantage), English as additional language support and special educational needs support are all much higher than the national Wales comparators. Findings, therefore, are very much localised to this particular institution.

5.3 Results and Discussion

The aim of the analyses were to identify gains in the aspects of literacy and computational thinking that were tested for following the intervention. The quantitative data analyses were carried out using IBM SPSS. A narrative and discussion of the findings is provided in this section. *Appendix D* contains both the full tabular outputs and SPSS data file upon which this discussion is based.

[1] See *Macefield, 2007* for more on the Hawthorne Effect phenomenon.

5.3.1 Comparison of Means

Between taking the pre-test and the post-test, learners who did not receive the microworld intervention improved by 1.9 marks on average in computational thinking (*see* Table 16). Learners who did receive the microworld intervention improved by 3.2 marks. Learners following a microworld-based teaching scheme demonstrated an improvement greater than the comparison group.

		Computational Thinking			Aspects of literacy		
		Pre-Test	**Post-Test**	**Gain**	**Pre-Test**	**Post-Test**	**Gain**
Experimental	**Mean**	13.1	16.2	**3.2**	20.8	24.7	**3.9**
	N	13	13	13	13	13	13
	Std. Deviation	4.3	4.1	2.7	3.1	3.8	4.4
Comparison	**Mean**	14.9	16.7	**1.9**	22.6	24.6	**2.1**
	N	14	14	14	14	14	14
	Std. Deviation	4.4	3.0	2.5	4.9	2.4	3.6

Table 16: WP2 Intervention Comparison of Means

For the literacy aspects, the comparison group improved by an average of 2.1 marks between taking the tests whereas the microworld group improved by an average of 3.9 marks (*see* Table 16).

This initial analysis is promising with a greater increase in performance for both the computational thinking and literacy aspects for those learners following the microworld-based teaching scheme. It is important to note, however, that the standard deviations are quite restricted in places (*see* Appendix D for tabulated outputs). A ceiling effect was perceived indicating a lack of sufficient range and that the tests may be too easy. In future experiments, there is a need to look again at the test designs used in order to ensure a more robust set of outcomes.

5.3.2 Significance Testing

Two independent samples t-tests were carried out in order to assess the confidence of the changes that were measured.

5.3.2.i Independent Samples T-tests

Both the change in computational thinking (p = 0.202) and aspects of literacy (p = 0.242) produced close to an 80% confidence level for change. This was particularly pleasing given the small sample size that makes achieving statistical significance difficult.

Further, Cohen's effect size value was calculated in order to measure the strength of the effects. Following the microworld intervention, a modest-to-moderate effect size (d = 0.50) was reported in computational thinking and a modest effect size (d = 0.46) in literacy. Potentially, then, the data reveals a positive effect of the microworld intervention in terms of both computational thinking and the literacy aspects that were tested for.

Gender did not feature as a guiding question in this research project, but t-tests comparing the gains in aspects of literacy and computational thinking grouped by gender were also carried out. The mean gain for boys was slightly higher than that seen by girls in both the computational thinking (+0.7) and literacy (+0.6) tests. The difference in performance was not statistically significant for either computational thinking (p = 0.657) or literacy (p = 0.570), which produced confidence levels of 34% and 43% respectively. With such a low level of confidence there appears to be little evidence of a tangible difference in performance between boys and girls.

5.3.2.ii Pearson's r

A subsidiary analysis was carried out in order to examine whether those participants who made the most improvements between taking the computational thinking tests (computational thinking gain) also made the most improvements in the literacy tests (literacy gain). This analysis incorporated results from both the microworld intervention group and the comparison group. To examine this relationship, the Pearson correlation coefficient was calculated.

The analysis revealed that there was a modest-to-moderate positive correlation (r = 0.341) between improvement in computational thinking and improvement in literacy. This finding is supported by a reasonable level of confidence that is greater than 90% (p = 0.082). These results suggest the possibility of a link between performance in these specific aspects of literacy and computational thinking. According to the results of the calculation, those who improved the most in the post-test for computational thinking also improved the most in the post-test for literacy.

5.3.3 Forming a Regression Model

A multiple linear regression model was used to examine whether (i) being part of the microworld intervention group and (ii) making a higher improvement in computational thinking served as combined predictors of higher performance in the literacy aspects.

The standardized coefficients (beta values) for the two variables individually reveal that making a high improvement in computational thinking (B = 0.301, p = 0.137)) seemed to have a stronger effect on the literacy aspects than being part of the microworld intervention group (B = 0.157, p = 0.431). The stronger comparative effect size when using computational thinking performance as a predictor is also matched with a higher level of confidence than membership of the intervention group.

These results suggest that making an improvement in computational thinking had a larger effect on improvement in literacy than following a microworlds-based teaching scheme. It is important to note, however, that the combined effect suggests a poor overall fit of the model to the data (R^2 = 0.139, adjusted R^2 = 0.068).

5.4 Conclusions and Limitations

The findings of the study reveal a number of important results for educators to consider. The data reveals that there are potential quantifiable gains in performance that could be achieved as a result of incorporating elements of a microworld-based pedagogy into classroom practice.

It has been revealed, with close to 80% confidence, that learners who received the microworld-based intervention in teaching practice made a higher improvement in computational thinking and specific aspects of literacy than their counterparts who did not. Further, the effect size of this was modest-to-moderate in both cases.

The study also reports, with 90% confidence, that there is a positive correlation between performance gains in computational thinking and gains in literacy.

Nevertheless, these findings are limited by a number of factors. The quasi-experimental design, though useful at embedding the findings within a classroom context, did not allow for random sampling. It is for this reason that a non-equivalent *comparison* group was used in lieu of a *control* group. Nevertheless, as the methodology states, every effort was made to control for external factors such as attainment level and scheme of work. The sample size was low for both the intervention (n = 14) and comparison (n = 13) groups. Despite this, it was decided to report confidence levels as there is evidence to suggest (see Muijs, 2011 for an explanation) that the t-test remains a robust instrument despite violations such as small sample size and non-random assignment.

The perceived correlation between computational thinking and literacy also may simply indicate that individuals tended to score high or low on both tests. It may be argued, therefore, that this does not provide a sufficient reason to interpret the correlation as a sign of a treatment effect.

Further, it is helpful to look at previous academic work to consider other possible limitations of the study. Papert, in *Microworlds* (1980a), claimed that learners could improve their problem solving skills by using his own Turtle Graphics microworld. Pea (1984) investigated this claim and suggested that it is the guidance provided by the enthusiastic teacher, not the microworld *per se*, that can lead to such an outcome. Papert (1987), in turn, argued that Pea's research focused too narrowly on specific indicators of problem solving, therefore omitting other improvements.

Pea's criticism and Papert's counter-criticism are equally relevant when considering the limitations of this study. First, the WP reported on an intervention led by one subject-specialist English teacher with an enthusiasm for the project. Selection of the subject-specialist teacher was carried out based on their willingness to participate. The English teacher that volunteered to participate did so due to as a result of a pre-existing interest in technology-enhanced literacy development. Gains in computational thinking and literacy, therefore, may be higher than what could be reasonably expected of other teachers. Conversely, the gain may also be quite understated due to the restrictions imposed by the test designs. It is important to note that the findings relate only to the specific elements of computational thinking and literacy that were tested for. Gains in other areas may have been made, though these were not tested for.

Selwyn (2014) warns academics that a more critical approach to educational technology is needed that moves beyond a 'disdain of formal education' and the assumption that 'education is best organized along informal lines of discovery, play and "hard fun"' (p. 161).[2] Tracing the routes of such tendencies in educational technology to Papert, Selwyn finds an apparent diminishing role of the teacher and classroom that is evident in his writing. What this WP has indicated, however, is that it may in fact be possible to build on Papert's work effectively. By implementing the microworld as a pedagogical tool in a formal classroom setting, a modest improvement in computational thinking and poetic thinking was recorded.

WP3 and WP4 replicates this research with different samples of learners and introduces an additional text-based programming component in Small Basic. There were two key problems that needed to be addressed. First, the standard deviations of the test scores indicated that the quantitative assessment instruments needed to be checked for suitability to ensure that a robust set of differentiated outcomes is achieved. Second, there is a need to report on the use of qualitative techniques in order to gain deeper insights into the microworlds-based teaching approach and any link that may exist between computational thinking and literacy.

[2] See Papert, 2008b for an account of the phrase 'hard fun' that Selwyn refers to here.

6 Work Package 3: Making Poems with the Block-based Snap! and the Text-based Small Basic Languages

6.1 Introduction

WP3 onwards moves back to School 1 where the teacher-researcher is employed as a teacher of ICT. It reports once again on a comparative evaluation of a microworlds-based approach to teaching poetic verse but within the subject domain of drama. This subject domain was chosen instead of English because (i) the English/Drama teacher at the school was willing and enthusiastic about participation and (ii) the activities could take place with a mixed-attainment group that is not streamed according to academic attainment. A quasi-experimental design was used in order to measure performance gains in specific aspects of computational thinking and literacy development following the microworld-based intervention.

The previous WP revealed a need for further qualitative data to provide a richer set of data about the microworlds-based approach. As such, this WP incorporates an analysis of survey data where learners reflect upon their own learning and the microworld activities that were completed. The restricted standard deviations with a ceiling effect in the previous WP meant that it was necessary to repeat quantitative data collection here with revisions to the test designs. It is important that a wider standard deviation is achieved to ensure that the test designs are robust and that findings are valid across attainment levels. It is possible that the tests were, put simply, 'too easy'. The pre-tests and post-tests in this study adopted a more formal exam-like structure and incorporated more challenging questions to address the full range of attainment levels. Further quantitative work was also needed as the previous WP took place with lower-attaining learners whereas School 1 provided the opportunity to carry out research with a mixed-attaining sample.

A problem for educators wishing to introduce learners to programming through a block-based approach is how best to manage the progression to a text-based programming language. A new text-based programming activity in Small Basic was used in WP3 alongside the same block-based Snap! activity that was used in WP2. The new text-based activity is explained further below.

6.1.1 Design of the Small Basic Microworld

The microworld created using Small Basic draws on the previous activity created in Snap! and then translates this to a text-based programming language. Learners interact with two different components of the Small Basic interface: the code editor window and the text window. This differs from the Snap! microworld where the learner primarily interacts with the scripting pane and views the sentences generated on the stage.

Learners begin using the code editor window. Here they set the labels for (i) different poetry forms and (ii) different word boxes. Note that the functionality of working with more than one poetry form at once is an additional feature that extended beyond the Snap!-based implementation. Learners were then required, using simple Small Basic code, to specify the sentence structures that are needed in each of the poetry forms.

The second part of the activity involves (i) adding words to the different boxes and (ii) generating poems using random picks from these boxes. An additional feature is the ability to generate different types of poem by selecting from the different poetry forms that were coded using the text window. These aspects of the activity take place when the program is run the text window.

Appendix B shows the code for the Small Basic microworld and the activity file created in Small Basic. A video demonstration is also provided here. The most important subroutines that the learners work with in the code editor and text window are reproduced and discussed here in the step-by-step order that they are encountered (*see* Table 17).

Step	Description	Mode of interaction	Subroutines used
1	Adding labels to the word boxes and the poetry forms	Code editor window	• BoxLabels • PoemLabels
2	Writing the sentence structures	Code editor window	• Poem#
3	Putting words in the different boxes	Text window	• AddItems
4	Generating sentences and substituting words	Text window	• MakePoems

Table 17: Subroutines in the Small Basic Microworld

6.1.1.i Step 1 – Adding labels to the word boxes and poetry forms

An important difference between the Snap! and Small Basic microworlds is that the latter enables learners to work with up to five different types of poem whereas the former enables only one poem structure to be specified at a given time. This subroutine is where learners specify labels for the five different poem types. An almost identical subroutine to specify the labels of the different boxes is called *BoxLabels* and is located in lines 12-19.

```
3          Sub PoemLabels
5            Poem1Label = "Haiku"
6            Poem2Label = "Tanka"
```

```
7     Poem3Label = "Cinquain"
8     Poem4Label = "Custom Poem 1"
9     Poem5Label = "Custom Poem 2"
10  EndSub
```

6.1.1.ii The Poem# Subroutine

Lines 21-27 shows how the learner structures each poem in the text editor. The example shown creates a haiku. Learners make a random pick from each box by typing *Box1Pick, Box2Pick,* etc. A key difference to the Snap! microworld is that each pick from a box here remains the same until the *Randomize* subroutine is called. This is shown in line 23.

```
21  Sub Poem1
22    'Pattern for the first poem type
23    Randomize()
24    TextWindow.WriteLine("On the " + Box2Pick + " " + Box3Pick)
25    TextWindow.WriteLine("Up where a " + Box1Pick + " " + Box4Pick + " " + Box5Pick)
26    TextWindow.WriteLine("Oh! " + Box2Pick + " " + Box4Pick + "!")
27  EndSub
```

6.1.1.iii The AddItems Subroutine

Learners encounter the *AddItems* subroutine (*see* lines 193-312, Appendix B) in the Text Window rather than the Code Editor. Learners type in their responses in order to insert words exemplifying the different word classes into the different boxes.

6.1.1.iv The MakePoems Subroutine

The *MakePoems* subroutine (lines 115-191, *see* ibid.) is also interacted with through the Text Window. Here learners use numeric responses to generate poems of the different types that were specified in the Code Editor.

6.2 Methods

To investigate the research questions within a realistic classroom context, once again a quasi-experimental nonequivalent comparison group design was used. Two year seven drama classes (ages eleven to twelve) were selected based upon the willingness of their subject-specialist teacher and learners within the classes to participate.

A subject-specialist English and drama teacher at the school led the teaching scheme. Drama was selected as a subject area instead of English in order to secure a more diverse attainment range.

Learners in English classes are streamed according to academic attainment levels but Drama classes are mixed attainment in their intake. This contrasts to WP2 where the sample consisted of learners from lower-attaining sets.

6.2.1 Design of the Teaching Scheme

Both groups followed the SoL unit for 'Year 7 Verse Drama' that was devised by the school. The SoL unit devised by the school aimed to teach about poetic verse over a period of six weeks.

In the first lesson, both groups were taught in a classroom about the concept of syllabic verse and were shown examples of poetry that subscribed to different fixed poetry forms such as haiku, tanka, etc. Learners were also asked to think about the classes of words that were used in these poems such as noun, verb, adjective, etc.

In this initial lesson the learners were also informed about the theme for their poetic verse project. The theme of the project was 'bullying'. Links were drawn to the school's PSE (Personal Social and Health Education) SoL. This in turn referenced the following aspects of active citizenship within the PSE framework for 7 to 19-year-olds in Wales: 'develop respect for themselves and others' and 'how to recognise and challenge effectively expressions of prejudice, racism and stereotyping' (DCELLS, 2008j, p. 20).

The second lesson for both groups was spent carrying out the pre-tests in computational thinking and literacy in a classroom.

The process of writing the fixed form poetry took place during lessons 3-5 of the teaching scheme and this was when the microworld-based intervention took place. The intervention group was taken to a bookable ICT suite for these lessons and spent two hours progressing through the steps for generating poetry with the Snap! microworld (*see* 5.1.1 for a description and Appendix D for the microworld activity). One lesson was spent generating poetry with the Small Basic microworld (see 6.1.1 for a description and Appendix B for the microworld activity).

Whole-class discussion and incidental questioning was used throughout to guide learners through the process of generating poetic verse with the microworlds. Regular differentiation was provided by support during these lessons. In the final fifteen minutes of the lesson five, the intervention group answered the questions contained in the microworld activities evaluation.

Lessons 3-5 for the comparison group consisted of learners devising fixed form poetry using an offline methods of flipchart paper and a pen in small groups in the drama studio. In line with the haiku form, learners devised three freeze frames or tableaux (*see* British Broadcasting Corporation, 2014 for an accessible guide to these exploratory terms) to perform the three lines of a haiku on the topic of bullying with one learner narrating and reading the poem aloud. This activity was chosen as it is closely aligned with established pedagogical practice within the subject area and therefore familiar to the teacher and learners. It is important to note that a more direct comparison would be the use of physical boxes to store words in categories in order to generate poem templates.

The final lesson for both groups was spent carrying out the post-tests in computational thinking and literacy in a classroom.

6.2.2 Design of the Pre-tests and Post-tests for the Quantitative Analysis

At the beginning of the half term, both the experimental (n = 25) and comparison (n = 26) groups were issued with paper-based computational thinking and literacy assessments. These were revised in light of concerns regarding the standard deviation of outcomes in WP2. The revised tests for WP3 can be viewed in *Appendix E.*

A more wide-ranging set of questions (in terms of attainment level) was used in the revised tests for WP3 in an attempt to secure a more diverse range of responses. An additional change was that a more standard testing format was used. The enhanced guidance provided in the previous tests (*see* Appendix D) was removed here.

	Q1, Q3, Q6, Q8	Q13	Q2, Q4, Q5, Q7, Q9, Q10, Q11, Q12
Adapt structures in writing for different contexts	X		
Use a wide range of sentence structures choosing connectives		X	
Use knowledge of word roots and families; grammar, sentence and whole-text structure.			X

Table 18: Literacy Assessment Matrix

A final step was taken to strengthen statistical validity by removing a potential confound between treatment and practice effects. Both pre-tests and post-tests were once again mapped identically, by question number, to the Welsh Government's Literacy and Numeracy Framework and Brennan and Resnick's computational thinking framework. The content within each question, however, was different. The tests in WP2 used exactly the same questions for the pre-tests and post-tests. The pre-tests and post-tests in WP3, though identical in their mapping to the computational thinking framework, were different in their content in order to strengthen the validity of outcomes.

Table 18 maps the questions in the literacy tests to the National Literacy and Numeracy framework expectation statements. *Table 19* maps the questions in the computational thinking tests to the computational thinking dimensions identified by Brennan and Resnick.

	Q1, Q4	**Q2**	**Q3**	**Q5**
Sequencing *(concept)*		X	X	
Conditionals *(concept)*				X
Testing and debugging *(practice)*	X			
Abstracting and modularizing *(practice)*				X

Table 19: Computational Thinking Assessment Matrix

6.2.3 Design of the Survey for the Qualitative Analysis

The digital tool Surveymonkey.com was used to capture survey responses for the qualitative analysis. The intervention group learners were provided with fifteen minutes of a timetabled session in a bookable computer room to complete the survey and were encouraged to provide full and detailed responses. This data was later exported and the extended written responses subjected to content analysis. Some quantitative data was also generated from the closed-style questions that were included.

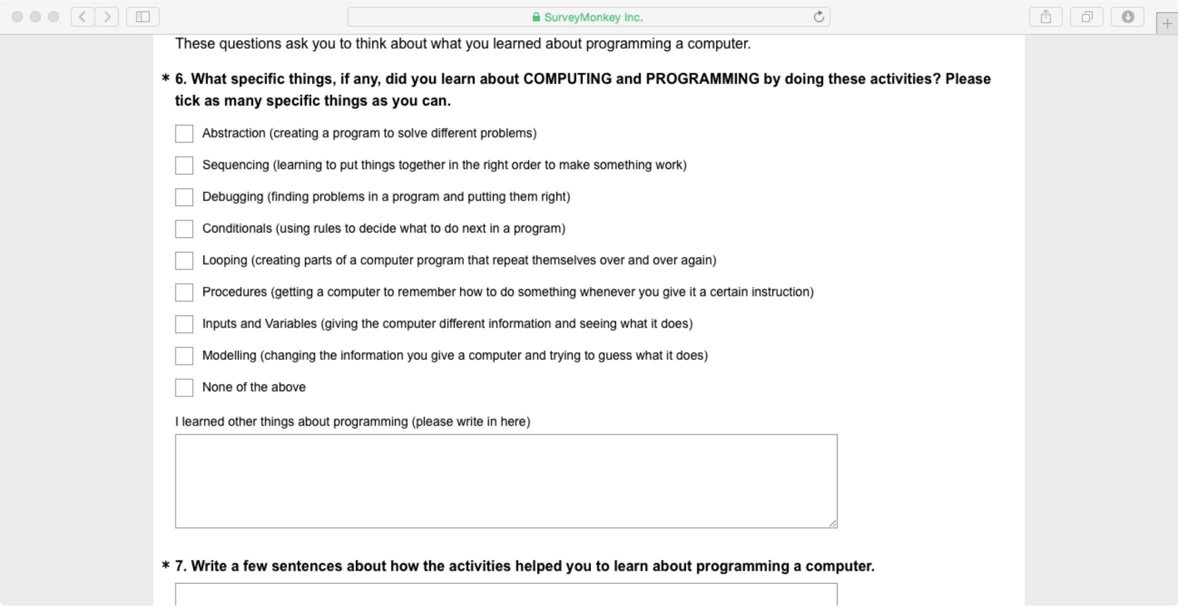
The survey was designed over several screens with questions being organized thematically. Some of the key screens are discussed in turn below.

6.2.3.i Computational Thinking Questions

In this screen (*Figure 13* provides an example), learners were first presented with a list of computational thinking dimensions in the form of check-boxes with captions explaining the meaning of each dimension. Examples include computational thinking *concepts,* such as sequencing and looping; and *practices*, for example debugging. Learners were able select no or multiple check-boxes in order to identify those areas where they feel an improvement had been made following the microworld intervention.

The use of a pre-determined checklist of responses was provided in order to help learners identify the areas where they have improved, though it is important to acknowledge the possibility of 'leading the thinking' of learners by taking this approach. To minimise this limitation, the self-reporting data was triangulated using quasi-experimental testing.

A text area was used, and a detailed response encouraged by the subject teacher, to enable learners who thought their computational thinking knowledge had improved to articulate in detail *why* they think this is the case.



These questions ask you to think about what you learned about programming a computer.

*** 6. What specific things, if any, did you learn about COMPUTING and PROGRAMMING by doing these activities? Please tick as many specific things as you can.**

□ Abstraction (creating a program to solve different problems)

□ Sequencing (learning to put things together in the right order to make something work)

□ Debugging (finding problems in a program and putting them right)

□ Conditionals (using rules to decide what to do next in a program)

□ Looping (creating parts of a computer program that repeat themselves over and over again)

□ Procedures (getting a computer to remember how to do something whenever you give it a certain instruction)

□ Inputs and Variables (giving the computer different information and seeing what it does)

□ Modelling (changing the information you give a computer and trying to guess what it does)

□ None of the above

I learned other things about programming (please write in here)

*** 7. Write a few sentences about how the activities helped you to learn about programming a computer.**

Figure 13: A Survey using Surveymonkey.com

6.2.3.ii Literacy Questions

Following this was an almost identical screen. This time it contained a series of questions regarding the particular aspects of literacy that were targeted by the microworld-based activities. The areas of focus were word classes, common poetry forms and the use of syllable and stress in constructing metre. These were referred to as 'areas of English' as it was thought that this term was more easily recognizable than the term 'literacy' within the school context. Learners were again encouraged to provide an extended response in a text area explaining their rationale behind these judgments.

6.2.3.iii Effective Practice Question

The final question probed learners for feedback regarding how the activities could be improved. This question was instrumental in providing qualitative data contributing to GQ three of the project its objective to establish effective practice recommendations for microworld pedagogic design.

6.3 Results and Discussion

The results begin with an analysis of the quantitative data before moving to an analysis of the qualitative survey data. *Appendix E* contains the SPSS data file, calculations syntax and full tabular outputs for the quantitative analysis.

6.3.1 Quantitative Results

Two independent samples t-tests were carried out to examine the effect of the microworld intervention upon the specific aspects of (i) literacy and (ii) computational thinking that were tested for.

The findings are limited by a number of factors and should therefore be treated with an appropriate level of caution:

(1) The subject-specialist teacher that carried out the intervention agreed to participate in the study as a result of her interest in technology-enhanced learning. This could have inflated gains.

(2) Drama is a mixed-attaining subject at the school and learners participated within their standard drama classes in order to retain a realistic educational setting. Variability of attainment level groupings in other subject areas, particularly that of English and ICT, were not controlled for.

(3) The results may simply indicate an improvement in taking these particular tests and not in specific aspects of literacy. In order to minimize the impact of this concern, the questions were carefully mapped to specific aspects of literacy and computational thinking.

6.3.1.i Literacy Improvement

The literacy tests were scored out of a maximum of 62 marks. The intervention group reported a mean difference of +6.8 marks between the post-test and the pre-test. The comparison group reported a mean difference of +1.5 marks between taking the tests (*see* Table 20).

An independent sample t-test reveals that, among the two drama classes taking the tests focusing on specific aspects of literacy, (N = 51), there was a statistically significant difference between the two groups, intervention (M = 6.8, SD = 8.0) and comparison (M = 1.5, SD = 8.8), t(49) = 2.24, p =

.029. Further, Cohen's effect size value (d = 0.63) suggested a moderate level of practical significance. This outcome suggests that the intervention had a moderate impact in raising attainment in the specific literacy expectation statements that were targeted within the National Literacy and Numeracy Framework.

	Group	N	Mean	Std. Deviation
Literacy Gain	Intervention	25	**6.8**	8.0
	Comparison	26	**1.5**	8.8
Computational Thinking Gain	Intervention	25	**.8**	14.7
	Comparison	26	**-7.7**	9.5

Table 20: WP3 Intervention Comparison of Means

6.3.1.ii Computational Thinking Improvement

The tests that targeted specific areas of computational thinking were scored out of 41 possible marks. The intervention group reported a difference of +0.8 marks between taking the tests at the beginning and end of the half term whereas for the comparison group this was -7.7.

The second independent sample t-test reveals that, among the two drama classes taking the tests focusing on specific aspects of computational thinking, (N = 51), there was a statistically significant difference between the intervention and comparison groups, intervention (M = 0.8, SD = 14.7) and comparison (M = -7.7, SD = 9.5), t(49) = 2.45, p = .018. Further, Cohen's effect size value (d = 0.68) suggested a moderate level of practical significance.

This finding for the comparison group is very surprising. The results show that the comparison group saw a drop in performance between taking the two tests that was much greater than the gain in performance reported by the intervention group. An investigation into the reason for this anomaly revealed that this was a result of an issue with the timing of the computational thinking post-test for the comparison group. This test was completed *after* the literacy test had been completed in the final lesson and the subject specialist teacher reported that this was rushed. Several learners in the comparison group independently confirmed that they 'ran out of time' when taking the computational thinking post-test. This is an issue that is addressed in WP4 with stricter attention paid to timings and test organisation.

In WP2 an issue arose with respect to the quantitative instruments producing a lower-than-expected standard deviation in outcomes. A ceiling effect in the standard deviation indicated that the test designs were too easy. Following revisions to the tests, this issued was rectified in WP3 where the standard deviations were higher in both the literacy and computational thinking tests.

Boys improved less than girls in both the literacy (-3.1 marks) and computational thinking (-2.2 marks) tests, but independent samples t-tests revealed no differences of statistical significance (p = 0.217 and p = 0.555). Though neither value is significant statistically, it is interesting that a higher confidence level was reported for the difference in performance in literacy. This means that it is more likely that gender differences affected outcomes in literacy than computational thinking.

6.3.1.iii The Relationship Between English and Computational Thinking

As with WP2, a linear regression model was used here to examine whether (i) high gains in the computational thinking test and (ii) receiving the microworld intervention combine to form an effective predictor of high scores in the literacy test. The results suggest there is a modest fit between the model and data (R^2 = 0.376, adjusted R^2 = 0.141).

There is some evidence to suggest, then, that achieving a high score in the computational thinking test and receiving the microworld intervention, when taken together, predicts a high score in the literacy test. The standardized coefficients reveal that gain in computational thinking (Beta = 0.228, p = 0.136) and being part of the intervention group (Beta = 0.232, p = 0.108) served as predictors of literacy gain to a near 90% confidence level of change. Though this evidence is not particularly strong, this finding suggests that there may be a link between high achievement in the concepts and practices of computational thinking and the aspects of literacy that were tested for.

6.3.1.iv Quantitative Survey Data

As part of the survey of intervention group respondents at the end of the study, learners were provided with a series of computational thinking concepts and asked to indicate areas where they feel an improvement had been made following the programming activities in Snap! and Small Basic. The results are presented in graphical form in *Figure 14*.

Many learners thought that their understanding of how to sequence a computer program had improved. A smaller majority of learners felt their knowledge of variables, debugging and abstraction had also improved. A minority felt that they had not developed their understanding of looping constructs even though looping constructs were engaged with in both activities. This is an interesting finding and may indicate that learners do not recognize the use of loops in their own computer programs. Further scaffolding and teacher intervention regarding this computational concept should clarify how this concept relates to the activity being undertaken in WP4.

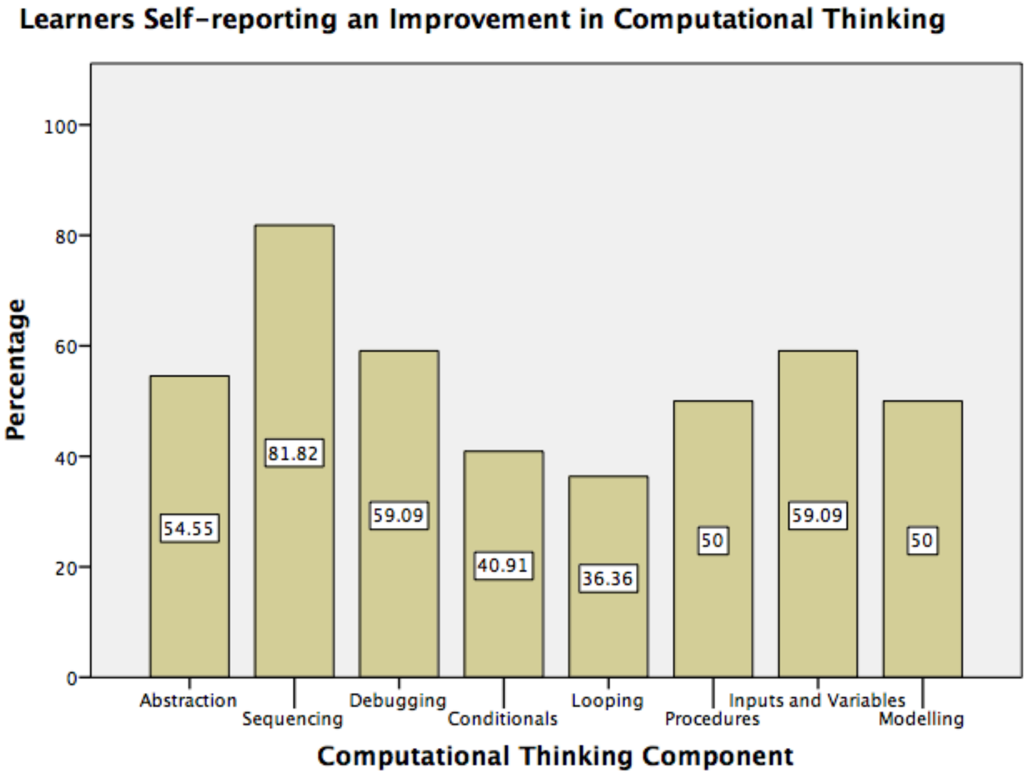


Figure 14: Computational Thinking Development Frequency Chart

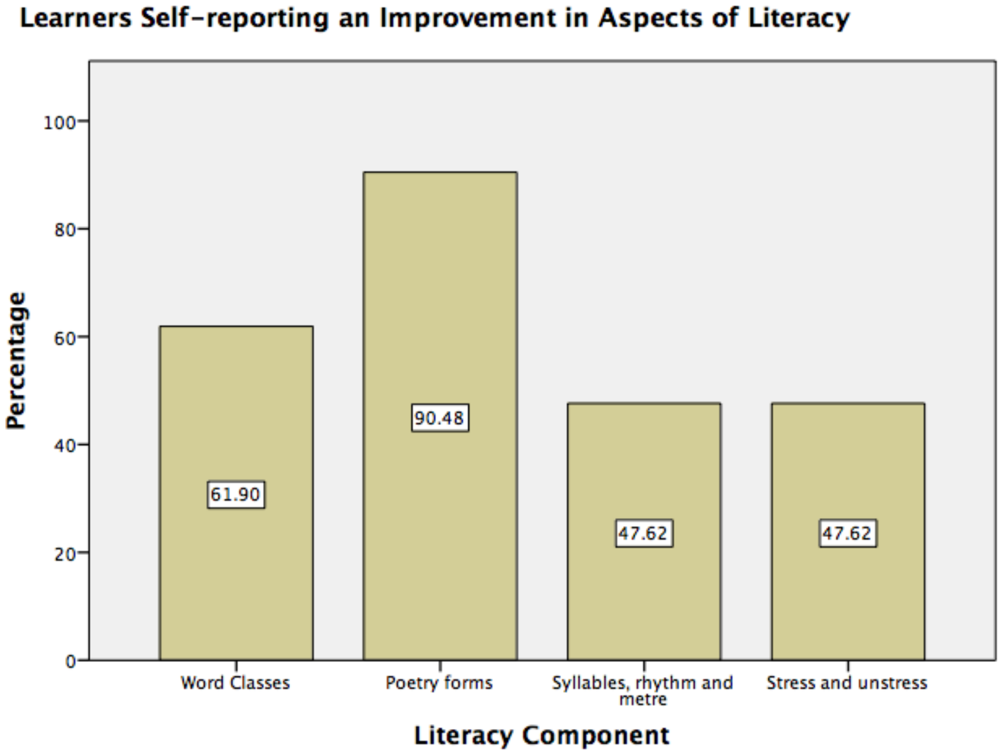


Figure 15: Literacy Development Frequency Chart

The survey also provided learners with a series of literacy concepts and asked them to select those areas that they feel has been further developed by using the two microworlds. The results are presented in graphical form in *figure 15*. Nearly all learners thought they had acquired a greater understanding of different poetry forms as a result of these activities. A smaller majority reported that this was also the case for recognizing word classes. Only a minority of learners felt that the activities improved their understanding of (i) syllable construction and (ii) stress. The latter is not surprising as this aspect of language is not strongly targeted by the design of the microworld. The finding pertaining to syllabic construction is more surprising as both the pre-test and post-test mapped questions to these specific areas of literacy and the resulting assessments revealed a marked improvement. This means that even if the learners felt they had not improved in this area, the analysis of outcomes suggests that they did in fact improve.

6.3.2 Qualitative Survey Data

In addition to the closed multiple-choice style questions, extended responses were encouraged through three text-area open questions. Learners were asked to provide a response outlining how the activities assisted literacy and computational thinking development. Further, learners were asked to consider how the microworld activities could be improved in order to increase impact. The content analysis of this data forms the structure of the qualitative findings.

6.3.2.i Literacy Development

First, the qualitative survey responses did not always seem consistent with the quantitative findings. When asked to articulate literacy improvement in extended text form, the majority of learners seemed to discuss improvements related to their understanding of different word classes. This is not consistent with the quantitative data where the largest majority of learners reported an improvement in their recognition of different poetry forms. Typical responses here seemed to link the activities to an improvement in their understanding of the different 'types' of words used when writing a poem.

> [T]he activities has helper [*sic*] me to understand the meaning of the different words and the poetic types of language. (Respondent 16)
>
> The activities helped me in English because I now know what and adjective, verb and adverb means without looking in my book or asking anyone. (Respondent 22)

It may also be the case that, specifically when related to word classes, that the activities are effective in consolidating existing knowledge in addition to introducing new knowledge for the first time. Respondent 25 comments that the activity helped her to 'understand *more* [*my italics*] about adjectives, nouns and verbs. The qualifier 'more' here appears to indicate a consolidation of a prior element of learning.

It is important to note, however, that not all learners were engaged by completing content that they had encountered previously. This was true particularly of participant fifteen, a more able and talented (MAT) learner: ‘I don't think it really made a difference on my learning or taught me anything I didn't already know. I honestly don't think it benefitted everybody.’ The sample (a mixed-attaining drama class) could not account for the variability in English sets and attainment levels amongst participants. As such, for a minority it did not provide enough literacy challenge.

6.3.2.ii Computational Thinking Development

When commenting on their development in computational thinking, several learners thought that undertaking the microworld activities aided their understanding of *how* a computer works beyond the experience of an end-user.

> [I]t helped me because I want a job in the computer industry and I need all of the things that you can get from a it lesson. its also helped me because if something happens on my computer at home or in school I can try to put it right. (Respondent 9)

It was interesting to find that one particular learner addressed the absence of programming in the school curriculum independently as a hobby outside of the formal learning environment. Responses such as this appeared to support the findings of the ICT Steering Group (2013) report that there is an absence of programming being taught by schools. The curriculum content at the school, mapped to the current National Curriculum (*see* DCELLS, 2008b) for ICT in Wales, provides only limited scope for programming activities.

One learner in particular welcomed the ‘original’ context of both activities, namely in constructing poetry. This perhaps reflects a recent proliferation of learning tools for introducing programming that is focused narrowly on video gaming. Websites such as Code.org for example, though potentially very useful, use popular video games like ‘Angry Birds’, ‘Flappy Bird’ and ‘Plants versus Zombies’ as the context for their programming tuition. Making poetry with a computer provided a different, and enjoyable, context with which to introduce programming.

> It helped me because I really enjoy programming and I do it at home. I use Python and Scratch but these activities have helped me to learn different programming languages. Also I am used to using programming to make games so I learnt in these activities to program poems. (Respondent 19)

A number of learners made links between learning programming and its applications in other subject areas. This is articulated in respondent 17’s comment that ‘this will also help in lessons like English and drama’. Learner 8 agreed that ‘it very helps because when you are learning IT you are also learning and English thats very helpful [*sic*]’.

Respondent 11 saw benefits that extended even further beyond subject-specific boundaries. He suggested that the cross-curricular skill of problem solving was particularly well-developed: ‘it helped me to learn to solve problems’. Respondent 12 appears to articulate this yet more deeply by focusing on a particular method of solving problems: 'It helped me learn by taking it step by step'. Learner 12 appears to reflect here on how the heuristic of sequencing figures as part of their own process of learning. This supports Wing (2006) who sees computational thinking as a tool for thought that transgresses subject-specific boundaries.

6.3.2.iii The Pedagogy of Microworlds

While the majority of learners recognized the benefits of the microworld activity, there were also several important comments to consider with regards to the pedagogy of a microworlds-based approach.

Whilst it is tempting for the teacher to progress immediately to launching the microworld, there was a clear need for more 'offline' non-computer-based activities to take place *prior* to the microworld being introduced. A possible way of achieving this is to use physical box equivalents for the word categories with pieces of paper representing the different words. This project did not explore this idea, but this is reported in Sharples (1985). Whereas there was a consensus that the activities were engaging, learners emphasized the need for further preparatory tasks to take place. Suggestions here ranged from a more in-depth classroom discussion to kinesthetic 'hands-on' preparatory activities.

> They could have explaimned it more clearly and fully check that we all understand. (Respondent 12)
>
> It could be changed by more thinking activities, and some writing but also could be more fun activities like using diferent shapes or other stuf to create something. (Respondent 8)

Whilst the process of programming was useful, some learners felt that the level of challenge inherent in the activities could be improved. Responses here were diverse and reflected the wide spectrum of attainment in the mixed-attaining class. Respondent 23, for example, commented that the activities 'could be a bit simpler because sometimes they are really confusing.' Respondent 16 provided a possible solution here by suggesting that learners should initially be provided with fully functioning poems containing errors. Learners then debug these poems before progressing to create their own poems. Introducing programming through the mechanism of debugging, therefore, may be a potential scaffolding strategy when working with lower-attaining learners.

> Maybe you can put the words they have put in the box it should make some sort of the poem and then the DE bug it then they can make a new one. (Respondent 16)

Respondent 19, on the other hand, explained how the activities should be made more challenging. An improvement could be made by 'teaching us how the […] blocks […] are made'. It is important to note, however, that the Snap! platform does enable learners to view the construction of the custom-built blocks. There is a potential for Snap! to be extensible, though the management of this as an extension task is more difficult in a realistic, time-limited classroom setting with a mix of attainment levels. Some learners, however, progress beyond *using* custom blocks by *creating their own* custom programming blocks in a later WP (*see* WP5).

One possible method of managing a mix of attainment levels is to employ a pair programming approach. The benefits of group work in education dates back to educationalists such as Vygotsky (1978) who proposed a dialectical mode of constructivism whereby more capable peers can provide support to those learners who require assistance. The benefits of pair programming are

The pair programming approach is useful because it stretches higher-attaining learners whilst providing support to lower-attaining learners. By acting as the 'navigator' of the programming process, the higher-attaining learner can verbally articulate their knowledge to help the understanding of the 'driver' who is typing the code (Plonka et al., 2011). Such an approach has well-documented benefits that include: (i) faster problem solving (*see* Cockburn and Williams, 2000), (ii) greater enjoyment of the coding process (*see* ibid.) and (iii) knowledge transfer through the act of learning from others (*see* Chong et al., 2005 and Resnick and Monroy-Hernández, 2008c). It is notable that two learners specifically referred to the possibility of a group work approach and its potential benefits to learning to be gained from social interaction. In WP4, it was decided to address the very problem of balance in the level of challenge by adjusting to a pair programming approach.

> Things that could be changed to make these activities better would be to do more group work on these subjects and to chat about the work more than usual. (Respondent 22)

Finally with respect to the pedagogy of microworlds, learners commented on what should happen *after* the poems have been generated using the computer. The focus of the intervention was largely on the *process* of generating poetry. A minority of learners commented on a need for a greater focus on *outcome*; that is the poems that are generated at the end. Learners had some interesting ideas here that including reading aloud – or even acting out – the poems that were generated. Again, there is research data by Resnick et al. (2008a) to suggest that learners benefit from the sharing of creative outcomes after programming projects are complete. As a result, learners were encouraged to read aloud their poems following the computer-based intervention when the sub-study was replicated in WP4.

> Maybe we could make the fun by reading out our poems and having feedback on them,i also think that we could act out some of our poems this would maybe help with class confidence. (Respondent 17)

Though it was not carried out as part of this project, an area for future research could focus on the possible merits of extending this to a performance within the wider school and community. A further possibility that was considered, though not carried out due to time constraints, was the publication of poetry in school publications such as newsletters.

6.3.2.iv Design of the Microworld Activities

The majority of comments made by learners referred to the pedagogy of the microworld-based activities rather than the design of the activities themselves. Nevertheless, there were some important design considerations for educators to consider when developing microworlds. The BYOB and Snap! platforms, sharing a block-based approach, can provide an accessible introduction to programming by reducing the problem of syntax that is experienced with text-based approaches. Despite this, there were criticisms made of these platforms. Interestingly, comments in this regard focused solely on the microworld in Snap! and not the microworld in Small Basic.

A practical concern that is shared by a minority of learners is the readability of the Snap! platform. Respondent 20, for example, explains that 'I think these activities could be made better by making the writing bigger so you can read it easily.' Snap! handles the large amount of text involved with this activity well, but the relatively small stage proved difficult for some. At the time of this study, there was no feature in Snap! that allowed the font size of the variable and list monitors to be increased. A 'zoom blocks' feature was added in a later update that provided a resolution to this issue in future WPs.

A further restriction of Snap! in comparison to Small Basic is the layout of the screen when undertaking the activities. Small Basic launches a new text window in the foreground that is detached from the code editor each time a program is run. With Snap!, on the other hand, the scripting pane remains on the screen each time a program is run unless a learner specifically pushes a control to make the stage full screen. Some learners felt that this layout caused confusion with respondent 18 explaining that 'certain parts of it [the Snap! microworld] were boring because they were complicated.' The Snap! microworld then, though potentially effective at introducing programming, is not without criticism as a choice of platform.

6.4 Conclusions and Limitations

The quantitative results in WP3 suggest that the learners who participated in the microworld-based teaching scheme saw a greater gain in computational thinking than their comparison group counterparts. This is consistent with the findings of WP2, though the confidence levels reported here are stronger. The survey data supports this evidence in that nearly all learners felt that their computational thinking skills had improved following the intervention. Further, in their responses several learners appreciated the value of programming as an activity and were keen to expand its presence in the curriculum.

The quantitative results also suggested that the microworld intervention led to a greater gain in the specific aspects of literacy that were tested for. This aspect of the quantitative work is also supported by the survey data with the majority of the learners self-reporting an improvement in their literacy following the microworld intervention. This was not true of all learners, however, with a minority of more able learners in the group feeling that they did not benefit greatly from the activities.

The regression analysis suggests that there may be a link between high performance in literacy and high performance in computational thinking. Though this link is stronger in this WP, it must be reiterated this relationship is a still a very modest one. This finding is also accompanied by some modest qualitative evidence to support the idea that there is a relationship between computational thinking and the specific aspects of literacy contained in the tasks. In their responses, some learners identified a link between programming and subject areas such as English.

There is, however, one surprising result. The comparison group showed a drop in computational thinking performance between taking the pre-test and the post-test result. This result is not consistent with WP2. Taking place at a different school, the previous study saw an improvement made by both groups with a larger improvement for the intervention group. The small sample size used here, along with a methodology that aspires to a realistic classroom setting, makes the study particularly susceptible to such threats to validity.

The subject-specialist teacher revealed that the reason for the unexpected drop in performance was an issue with pacing in the post-test lesson. Since the computational thinking tests took place subsequent to the literacy tests in the post-test lesson, time (specifically the lack of it as reported by the subject-specialist teacher) may be a prominent factor in this apparent anomalous result. Further investigation by talking to the comparison group learner participants confirmed that they had ‘run out of time’ before taking the computational thinking post-test. When replicated in WP4, therefore, this study will need to apply more robust controls during testing. This should take the form of stricter intermediate timings between each test and a closer monitoring of learners.

The qualitative findings also provide three further factors to guide movement into the next WP.

First, it was clear that more thought needed to be given to the structure of the intervention. Learners felt that more 'offline' or 'unplugged' guidance is required before starting to generate poems on the computer. Further whole-class discussion is necessary prior to undertaking the activities and resources will be developed to support unplugged activities prior to exploring within the microworld.

Second, learners indicated that there seemed to be too much focus given to the *process* of making a poem with little consideration given to the final result, the poems themselves. The teaching scheme of the intervention, therefore, needs to be refined to ensure that an appropriate teaching context is given to the microworld activities. WP4 will make use of a design exhibition format (*see* Resnick, et al., 2005) to enable the sharing of the poems that have been devised with the rest of the class.

Third, the pedagogy of the intervention needs to be adapted in order to reflect the full spectrum of attainment levels. Not only did a minority of more able students find that the activities lacked sufficient challenge, but a minority of the less able students thought that they lacked accessibility. WP4, therefore, moves to a pair programming pedagogy to encourage peer teaching. There is evidence to suggest that pair programming enables the co-construction of knowledge to take place through one learner taking the role of the 'navigator' and the second taking the role of the 'driver' (see Chong et al., 2005 and Plonka et al., 2011). Peer teaching can work to extend the higher-attaining by asking them to verbally articulate their own learning whilst also providing social scaffolding for the lower-attaining through the one-to-one assistance provided by a peer. Indeed, several learners identify the merits of a group-based approach in their survey responses. Such an approach also requires a new qualitative instrument: WP4 uses voice recording of the pair programming interactions in order to provide a richness to the qualitative analysis.

7 Work Package 4: Paired Programming with Snap! and Developing a Microworlds-based Teaching Scheme for Literacy and Programming

7.1 Introduction

The project as a whole, and each of the WPs within it, is characterized by an action research approach. In the words of Kemmis and McTaggart (2000), the researcher can 'observe the process and consequences' of a change and then move to '[reflect] on these processes and consequences and then replanning' (p. 595). There is much replication in this methodology between WP3 and WP4. Some key methodological changes implemented in response the findings of WP3 are discussed in section 7.2.

7.2 Methods

The intervention took place with two mixed-attaining year seven classes over a seven-week period. The classes were different to those who participated in WP3. The teaching scheme used an identical format to that described in *section 6.2.1*, but with some important differences arising from the action research cycle and evaluation of the previous study.

7.2.1 Design of the Teaching Scheme

The first lesson of the WP3 teaching scheme saw learners taught about concepts such as syllabic verse, fixed form poetry and stress for one lesson. In this WP4, two lessons of the extended seven-week program were dedicated to 'offline' or 'unplugged' activities that took place away from the computer. Close attention was paid to the structure of these lessons and several new teaching and learning resources were created.

Learners in both the intervention and comparison groups were required to complete a series of practice worksheets in literacy and computational thinking that included examples of fixed form poetry. The role of the teacher here was to (i) lead whole-class discussions of the different computational thinking and literacy concepts and (ii) differentiate support levels as appropriate whilst learners completed the worksheet activities. The first lesson focused on five areas of computational thinking: abstraction, sequencing, debugging, conditionals and general programming concepts. The second lesson focused on five aspects of literacy: syllabic form, stress, word class, poetry forms and poetry analysis.

The practice worksheets adapted the layout of the old-style, unrevised tests that were used in WP2. The unplugged worksheets were accompanied by new teaching aids in the form of slide-based presentations to facilitate whole-class discussion. The new teaching and learning resources created for WP4 can be viewed in *Appendix F*. It may be argued that unplugged activities are effective here because they serve to build a base level of computational thinking in order for the psychological state of flow to be achieved when moving to explore in the microworld (*see* section 2.4.2 and Csikzentmihalyi, 2004). These pedagogical changes are also supported by Ennis' (1992) assertion that direct teacher instruction is required in order to see an improvement in problem solving and thinking skills when undertaking programming exercises.

The third and seventh lessons for both groups were used to carry out the pre-tests and post-tests. The pre-tests and post-tests used were identical to those used in WP3 (*see* the tests in Appendix E). The quantitative data analysis of the previous WP revealed an anomalous result that was inconsistent with previous sub-studies. As a reminder, a negative mean gain in computational thinking was reported for the comparison group. Discussion with the subject-specialist revealed that this outcome was a result of a pacing issue in the post-test session for the comparison group with learners running out of time during the second computational thinking test. The teacher was responsible for enforcing stricter timings of thirty minutes per test in order to ensure that learners had an equal amount of time to complete both tests. These additional controls were put in place for the pre-tests and post-tests with both groups.

The process of constructing poetry took place during lessons 4 – 6 of the teaching scheme and it is here where there is a departure for the intervention and comparison groups. Both sets of learners were introduced to the haiku poetry form and were briefed to generate haikus. As with WP3, the haikus generated were required to link in some way to the PSHE topic of bullying.

The comparison group devised their haikus in small groups in the drama studio. The teacher provided flipchart paper and pens for the process of writing the haikus. In small groups, learners devised a script for the narrator to read aloud whilst three freeze frames were performed by the remainder of the group. The teacher circulated regularly during the devising process in order to assist with the development of ideas and used a spotlighting teaching technique in the final lesson to showcase the poetry performances devised by the groups.

The intervention group, on the other hand, spent the time constructing their haikus using the microworld-based activity. The teacher took learners to a bookable ICT room for these lessons.

The first lesson was largely dedicated to populating the word boxes with vocabulary related to the PSHE topic of bullying whilst the second and third lessons were dedicated to writing the poetry forms and generating the poems. The teacher was responsible for using whole-class discussion and incidental questioning prior to introducing these two phases of developing the poems. An exemplar completed version of the microworld was provided to help with modelling of outcomes. Learners were encouraged to adapt the exemplar before moving to a 'blank' version of the microworld containing only the custom blocks. Regular coaching was provided by the teacher during these lessons to offer differentiation to less able students.

A refinement made to the teaching scheme as a result of the qualitative data analysis in WP3 was an increased focus on the finished poems *after* they were generated. Pairs of learners were required to recite their poems to their peers during a fifteen-minute plenary session led by the teacher. Previously, a much greater focus was given to the *process* of programming with little opportunity for *showcasing* the results of learner work. This change is supported by a recommendation made by Resnick et al. (2008a) that learners should be given an opportunity to exhibit and share their creative outcomes after the process of coding is complete.

7.2.2 Peer Talk Analysis

Pair programming was introduced for the first time in WP4 as a result of recommendations made in WP3. Research suggests that pair programming can have many benefits including faster solving of problems and a greater enjoyment of learning (*see* Cockburn and Williams, 2000). There is also evidence to suggest that working in pairs to solve a programming problem can lead to participants acquiring the technical skills that are demonstrated to them by their collaborator (*see* Chong et al., 2005). The pairings were selected by the learner participants themselves. The peer talk analysis provides evidence that the pairings led to exploratory talk, and it is possible that the friendship groups played a positive role in encouraging an 'openness' to the dialogue. This is speculative, however, as no data was collected on the impact of the pairings upon the findings. An interesting avenue for future research would be an investigation of any differences in performance arising from teacher assignment based on factors such as gender or attainment level.

In order to collect qualitative data derived from the pair programming, it was decided to use audio recording as a data collection strategy as this enables a greater richness to surface when reporting findings instead of relying on field notes alone. Six pairs were selected according to their willingness to participate. Three pairs were recorded using the Snap! microworld and three using the Small Basic microworld, each for around ten minutes at a time.

Audio recordings were transcribed and analysed according to a coding scheme of three categories. These were based on a framework by Mercer and Wegerif (1997): (i) *disputational* talk, characterized by discord and individualized decision making; (ii) *cumulative* talk, characterized by a building upon the ideas of the collaborator; and (iii) *exploratory* talk. It is the latter type of talk, according to such a framework, that confers most clearly the process of reasoning and is characterized by critical and constructive exchanges between collaborators. It was hoped, therefore, that the audio recording data would reveal exploratory talk in exchanges between the pairs of learners.

7.2.3 Pre-tests and Post-tests

Quantifiable improvements in computational thinking and aspects of literacy were measured using the identical pre-tests/post-tests as the previous WP (*see* Appendix E for a reminder). The testing schedule was comprised of two thirty-minute pre-tests at the beginning of the half term and two thirty-minute post-tests at the end of the half term. Once again one of these targeted aspects of computational thinking and the other targeted aspects of literacy. Two classes formed the experimental (n = 21) and comparison (n = 21) groups. No learners in either class had participated in previous studies related to this project. Due to the nature of opportunity sampling and timetabling constraints at the school, the sample for WP4 was slightly larger than that for the prior WPs.

The tests were again mapped by question number to the Welsh Government's Literacy and Numeracy Framework and Brennan and Resnick's computational thinking framework. By comparing responses to each question, it was possible to isolate specific skill areas in literacy and computational thinking. Performance gains in questions relating to debugging, for example, could be compared to performance gains in other computational thinking areas such as sequencing. This enabled a fine-grained set of performance data on outcomes to surface.

7.3 Results and Discussion

7.3.1 Quantitative Results

As with the previous WP, two independent sample t-tests were carried out in order to quantify improvements in the aspects of computational thinking and literacy that were targeted by the tests.

7.3.1.i Overview results

The pre-tests and post-tests revealed a statistically significant difference between the two groups in computational thinking (p = 0.012) and an improvement in literacy (p = 0.092) that was close to a 90% confidence level for change. Cohen's effect size value for literacy (d = 0.54) and computational thinking (d = 0.83) describe a moderate effect of the intervention upon the areas of literacy - and a large effect upon the areas of computational thinking - that were tested for. A comparison of the gains can be viewed in *Table 21.*

	Group	N	Mean	Std. Deviation
Literacy gain	Intervention	21	**8.2**	8.7
	Comparison	21	**2.6**	12.2
Computational thinking gain	Intervention	21	**10.7**	10.7
	Comparison	21	**3.6**	6.4

Table 21: WP4 Comparison of Means

These findings provide further evidence to support the hypothesis that the microworld intervention correlates with greater gains in the aspects of computational thinking and literacy that were targeted. In the previous WP, the quantitative findings revealed a drop in performance for the comparison group between taking the pre-test and the post-test. That finding contradicted previous WPs and was thought to be an anomalous result. The additional controls placed on the timings of test-taking appeared to remove this anomaly as the comparison group saw a rise in computational thinking between taking the tests, though this was significantly lower than that demonstrated by the intervention group. This result is broadly in line with what would be expected.

Differences in gender performance in WP4 were fairly consistent with that reported in WP3. In WP3, boys made a smaller improvement in both literacy and computational thinking than girls. In WP4, a similar result was seen for literacy (-4.6), though boys improved slightly more than girls in computational thinking (+1.7). The probability measures arising from independent samples t-tests were similar to that seen in WP3. For literacy this was p = 0.198 in WP4 compared to the value of p = 0.217 in WP3. For computational thinking this was p = 0.571 in WP4 compared to p = 0.555 in WP3. Once again it appears more likely that any difference arising from gender would affect performance in literacy rather than computational thinking. Both analyses have produced results, however, that are not statistically significant. The results therefore need to be treated with an appropriate level of academic caution.

7.3.1.ii Literacy gains by question type

A unique feature of the quantitative analysis of this WP was that the pre-test and post-test marks were recorded at the level of response to each individual question. It was possible therefore to deconstruct the comparative gains made in literacy according to question type. The questions in the literacy tests were once again mapped to the following expectation statements of the National Literacy and Numeracy Framework (DfES, 2013):

- Adapt structures in writing for different contexts
- Use a wide range of sentence structures choosing connectives
- Use knowledge of word roots and families; grammar, sentence and whole-text structure.

The smallest difference in gain between the two groups was seen for questions where learners were required to identify different sentence structures. The difference here was 0.43 marks and this question type also returned the weakest confidence level (p = 0.701). The improvement in learners recognizing different word classes was much stronger with a mean difference of 2.62 marks and a confidence level of around 85% (p = 0.153).

A statistically significant gain (p = 0.047) of 2.57 marks on average, however, was reported for those questions where learners were required to write their own fixed form poem. In these questions, learners were asked to construct their own poem whilst adapting their writing to fit within the syllabic constraints of a given poetic form. These results are promising in that they appear to suggest a link between microworld-based pedagogy and an improvement in writing style adaptation to fit particular fixed poetic forms.

7.3.1.iii Computational thinking gains by question type

The computational thinking tests were also deconstructed according to question type. Resnick and Brennan's (2012) framework for computational thinking development with building block programming was used in order to classify the different questions according to the following concepts and practices:

- Sequencing (concept)
- Conditionals (concept)
- Testing and debugging (practice)
- Abstracting and modularizing (practice)

The area of computational thinking where the smallest improvement was made relates to the concept of conditionals with an average gain of 0.8 marks and a confidence level of 88% (p = 0.123). Gains related to the practices of testing and debugging (1.62 marks) and abstracting and modularizing (2.57 marks) were better with confidence levels of 83% (p = 0.167) and 84% (p = 0.155) respectively.

It was the computational thinking concept of sequencing where the most promising result can be seen. The gain reported here (2.19 marks) was not as high as that seen with questions related to the practice of abstracting and modularizing. The gain, however, achieved a confidence value of 92% (p = 0.079) that was stronger than that seen by the other question types. It may be the case, therefore, that the activities helped to develop a better understanding of how to sequence a computer program. This is potentially a positive finding: sequencing is a key element of programming that is required during later phases of the education cycle.

7.3.2 Qualitative Results

In order to analyse the voice recording data collected during the Snap! and Small Basic paired programming activities, Wegerif and Mercer's (1997) framework for the study of peer talk was used. A series of ten-minute voice recordings were made with different pairs in the class at key points during the intervention lessons. Learners were made aware that that the recordings would be transcribed and then immediately deleted. Learners were also informed that all reporting would be anonymous. The voice recordings were analysed for examples of three educationally significant types of talk according to Wegerif and Mercer's framework. Each of these will now be considered each in turn.

7.3.2.i Disputational Talk Sequences

According to the framework, **disputational** talk is found where collaborators learn from each other's mistakes accompanied a dialogue that is characterized by disagreement and individualized decision-making. The sequence below from the pair Ellie and Zac illustrates this:

Ellie: One syllable adj … what's a one syllable adjective?
Zac: *Fat*
Ellie: *Fat*. Yeah they could be bullied.
Zac: *Far*. You've put *far*.
Ellie: It won't let you go back! It's really silly!
Zac: Lemme ask Sir, a minute…. Get some help! Get some help!
Ellie: Oh my God, what did I do?
Zac: You've deleted it! You've deleted it!

Zac and Ellie were populating the boxes with examples of different word classes. Initially the dialogue shows characteristics of cumulative talk as Zac's example of the one syllable adjective 'fat' is embraced by Ellie who seeks to build upon this to construct a wider theme of 'bullying' for the poem. Things start to go wrong, however, when Ellie makes a simple transcription error and types the word 'far' instead. Talk quickly turns disputational as Zac distances himself from Ellie's pressing of the enter key and subsequent closing of the text window in a state of panic. The sequence shows how both learners have realized a constraint of the Small Basic microworld, namely that only new words can be added with no facility for amending the existing words already in the boxes. This learning point was achieved together through collaboration, but Zac's distancing himself from Ellie's mistake is characteristically disputational through 'disagreement and individualized decision making' (*ibid*., p. 53).

A further sequence illustrating disputational talk is seen in the exchange between Shona and Cariad. Cariad begins by suggesting that the words 'cat' and 'hippo' are nouns with one syllable:

Shona: One syllable – one syllable noun
Cariad: Hmmm – *cat*, *hippo*
Shona: *Hippo*?
Cariad: That's two!
Shona: Oh Yeah!
Cariad: What are we gonna do next?
Shona: Right what are we gonna do – hmmm – *girl* – And – Lets do *boy*.
Cariad: How [*sarcastically*] creative.

Shona reminds Cariad that she has not taken into account the syllable count and Shona agrees with this. At this stage, the sequence appears exploratory as Cariad's critical challenge appears co-operative rather than individualistic. This very quickly transitions into disputational form, however, when Shona suggests the words 'girl' and 'boy' should be added to the box. Cariad's sarcastic response to the suggestion runs in opposition to Shona's suggestion. Cariad criticizes the lack of creativity shown by Shona and does not offer any solutions.

7.3.2.ii Cumulative Talk Sequences

Cumulative interactions are those where learners attempt to maintain a sense of harmony between group members with little critical engagement. Ben and Matthew's sequence below illustrates cumulative talk through the mechanism of turn taking that is introduced by Ben.

Ben: Can I give you a noun? Whats a noun? Is it a name? Name or place…
Matthew: Yeah.
Ben: It is a name or place innit?
Matthew: Yeah
Ben: K. I'll do two, you do three – *James*.
Matthew: Na I think that's two syllables – *James* (*single clap*) – na, it's not.
Ben: *Elliot*.
Matthew: Na I think that's two syllables. *El* (*clap*) – *lee* (*clap*) – *ut* (*clap*)… Three.
Ben: Oohhh… *James* (*clap*). So if it was you it would be *Math* (*clap*) *hew* (*clap*)
Matthew: Yeah - *Math* (*clap*) *hew*.
Ben: *Dies* (*clap*) *el* (*clap*) – *Ben* (*clap*) – *Dan* (*clap*) *ny* (*clap*).
Ben: Ok… *Jack* (*clap*).

When arriving at a list of words to populate the one syllable noun box, Ben suggests that one possible approach is to divide the task into two areas of responsibility with Ben suggesting two nouns to contribute and Matthew suggesting a further three. This exchange does not display critical engagement but the mechanism of turn-taking works to successfully secure a sense of cohesion.

The initial cumulative talk by Ben and Matthew transitions, however, to incorporate critical engagement. Matthew initially claims that the noun 'James' is comprised of two syllables but uses the mechanism of clapping out loud to correct himself. This provides an example of an internalized pedagogical method for arriving at the correct number of syllables. It appears that Ben has not yet grasped the concept of syllable count when he suggests 'Elliot'. Matthew, who again uses the mechanism of clapping out loud to illustrate syllable count, shows Ben why this particular word would not fit. Ben, exclaiming 'ohhh' in a moment of realization, progresses to clap aloud a series of other nouns – each time seeking affirmation from Matthew – in order to eventually arrive at a correct suggestion of 'Jack'.

Matthew's encouragement as Ben learns to clap syllable structure out loud is a cumulative response that serves to strengthen cohesion amongst the pair. The exchange progresses beyond cumulative talk when Matthew offers constructive criticism when Ben makes a mistake with the suggestion of 'Elliot'. Matthew's challenge was not disputational and individualistic in nature. Instead the talk sequence reveals a supportive means of building knowledge in his partner. What we see here, then, is the transition to a different type of **exploratory** talk that Mercer and Wegerif suggest has a 'special status' as a 'situated model of reason' (*ibid*, p. 59).

7.3.2.iii Exploratory Talk Sequences

Exploratory talk sequences, according to Wegerif and Mercer's (1997) framework, are more complex interactions than those seen in other forms of talk. Disputational talk is characterized by competition with individualized decision-making. Cumulative talk is defined as a cohesive mode of interaction with harmony between group members. Exploratory talk is unique in that it combines elements of both former modes of interaction: 'critical challenges are supported but are contained within a co-operative framework' (*ibid.*). Exploratory talk enables a collaborative construction of knowledge that is advanced by constructive challenge:

> [Exploratory talk] allow[s] different voices to inter-animate each other in a way which not only constructs shared knowledge but also critically assesses the quality of that knowledge (*ibid*, p. 59).

The sequence by Scarlet and Morgan below illustrates how exploratory talk grew out of the microworld activity. Morgan is confused from the outset as to what characterizes the verb word class and seeks assistance from Scarlet who happily explains that they are 'doing words'. The sequence begins characteristically cumulative in nature as Scarlet simply ignores Morgan's mistaken suggestions of 'skip' and 'run' as they are not in the past tense. When Scarlet suggests 'jumped', something that causes confusion for Morgan who mistakenly identifies the word as an adjective, we see a transition in the talk sequence.

Scarlet reminds Morgan that 'jumped' is simply a verb in the past tense and is therefore not an adjective. Morgan's new-found understanding as a result of Scarlet's challenge is expressed through his exclamation of ‘woah!’. He ends by confirming with repetition that Scarlet’s word choice is valid. Scarlet’s constructive challenge has addressed a shortfall in Morgan’s knowledge without resorting to the disputational characteristics of confrontation and competition.

Scarlet: Errr – one syllable verbs in the past tense….
Morgan: What is a verb?
Scarlet: A verb?
Morgan: Yeah
Scarlet: Like – umm – a doing word
Morgan: *Skip, run*
Scarlet: *Jumped*
Morgan: *Jumped* is a describing word
Scarlet: No, this is past tense
Morgan: One syllable verb in the – woah – *jumped*.

What the latter two transcriptions illustrate is that the microworld activity is sufficiently complex to enable learners to move beyond what Mercer and Wegerif describe as the ‘very basic structural possibilities’ of disputational and cumulative talk to the more complex form of exploratory talk that incorporates ‘elements of both’ (*ibid*, p. 61). This progression from disputational/cumulative talk to exploratory talk may indicate that the microworld is effective at incorporating a ‘low floor’ and ‘high ceiling’ of difficulty (*see* Rieber, 2009).

Using the framework as a tool for analyzing classroom discourse has revealed that the microworld activity has the potential to incite complex verbal interactions. There is evidence here to suggest, then, that the microworld-based activity may prove effective as a strategy for stimulating shared knowledge construction when used alongside a pair programming pedagogy. It is not clear, however, to what extent the examples of exploratory talk arose here due to the complexity of the microworld. The exploratory talk may be an artefact of the pair working pedagogy rather than the design of the microworld itself.

7.4 Conclusions and Limitations

The quantitative results of this WP have supported and built upon the findings of the previous sub-studies. In terms of probabilities, the t-test for computational thinking revealed a statistically significant difference in gain for the intervention group following the microworld-based teaching scheme. A 90% confidence level of change was reported in the t-test for literacy. The Cohen’s D effect size value suggests a moderate strength of impact upon gain in literacy and a large effect on gain in computational thinking.

The quantitative analysis also appears to support the suggestion that the previous negative gain seen by the comparison group for computational thinking was anomalous. The additional controls on test taking removed this irregularity that surfaced in the previous WP.

One of the unique features of this WP was the reporting of pre-test and post-test results by question score rather than overall test score. This enabled new insights through an additional layer of quantitative data.

The new data revealed that the computational thinking concept of sequencing was particularly well developed by the microworld activities. This is perhaps what would be expected since a key process when using the Snap!-based microworld is placing blocks in the correct sequence in order to generate the desired poem. Though this reason is useful, the conditions of the research (where there was limited time spent was spent with learners and a lack of continuity) makes this link difficult to ascertain.

It is interesting to note that the intervention seemed to have a much more measured impact on the learner's ability to identify conditional or branching structures following the intervention. Again, this is perhaps not surprising given the design of the microworld activities: learners were not required to make use of conditional structures when generating their own poems. A future development, for instance, could involve the selection of either the word 'a' or 'an' depending on whether the initial letter in the subsequent word is a vowel or consonant.

In terms of literacy, a statistically significant improvement was seen for questions where learners were required to adapt their writing to fit a particular fixed poetic form (*see*, for instance, question 13 in the literacy tests of Appendix E). This appears to suggest a link between undertaking the microworld activity and an improvement in understanding the structural composition of fixed form poetry. There was a much smaller, though still positive, gain reported for questions that required learners to identify the composition of sentences in terms of their different word classes. It appears from these results, therefore, that learners, saw a greater improvement in questions requiring them to write their own fixed form poems in comparison to comprehension/reading style questions of existing poems.

The quantitative results, though positive and replicated across different sub-studies, should be treated with an appropriate level of caution. It is important to note that two out of three packages with quantitative elements made use of samples derived from a single school population. The computational thinking and literacy tests, in addition to the microworld activities themselves, also targeted very specific aspects of computational thinking and literacy development. It is important, therefore, to qualify these improvements by re-stating these limitations on validity.

Another unique feature of this WP was the discourse analysis of the pair programming talk that took place. A pre-defined coding scheme was used here according to Wegerif and Mercer's

(1997) framework. This offered a different approach to qualitative data analysis in comparison to the grounded theory model used in WP3. The analysis provides some evidence to suggest that the microworld activity stimulated a complex type of talk in the classroom that is exploratory in nature. According to Wegerif and Mercer's framework, exploratory talk provides the most optimum conditions for shared knowledge construction as it provides a model for reasoning. In exploratory talk, 'knowledge is made more publicly accountable and reasoning is more visible in the talk' (*ibid*, p. 54). The classroom discourse analysis of the paired programming provided examples of where the microworld was able to ellicit complex peer talk.

There is an inherent difficulty in establishing the extent to which the peer talk recorded was a result of the complexity of the microworld design or the paired working. An element of exploratory talk would likely arise with the introduction of any new resource in a classroom. The Snap! microworld, with its focus on computational thinking within a different curriculum area, may provide a particularly effective teaching context for exploratory talk given the novelty of the approach.

8 Work Package 5: Bringing Literacy to an extracurricular programming poetry club

8.1 Introduction

The qualitative dimension to WP4 provides modest evidence to suggest that the microworld-based activities in Snap! and Small Basic incite exploratory talk in the classroom. This is important because, according to Wegerif and Mercer's (1997) framework, exchanges of this type are integral to the process of constructing shared knowledge within a pair programming pedagogical orientation.

A particular problem caused by research embedded in real school settings is the constraints that curriculum organization places upon the research process. There was a concern that it is not possible, either through discrete English/Drama or ICT lessons, to fully exploit the microworld activities to their full potential. Due to constraints placed by schemes of learning, it would not have been possible to spend an extended period of time with a group of learners beyond one half term. In order to achieve a richer set of qualitative data, an extracurricular context was required.

This extracurricular club was promoted during assemblies and form tutor sessions to all learners in year seven (aged eleven to twelve) at the school. Learners showing an interest attended on a voluntary basis during lunchtime sessions. There is research coverage showing the merits of introducing programming outside of formal learning experiences, the largest of which is perhaps the Computer Clubhouse network by MIT (*see* Kafai et al., 2007c; MIT Media Lab, 2012a).

The club ran over the course of one full term of around twelve weeks during the academic year. Attendance at the club was variable with around six-to-ten students attending each session. The learners who attended remained relatively constant from the beginning.

There was no quantitative data to collect, no administration of pre/post-tests and no restrictions by scheme of work. The aim was that this freedom enabled learners to fully engage with the activities over a more substantive period of time. As a result, it was hoped this extended study would provide a new richness of qualitative data to surface.

There are many parallels that may be drawn between the work carried out in this WP and that reported by Sharples (1985). There are, however, some key differences that extend beyond a reimagining of these ideas for a block-based implementation. A key methodological feature that is distinct to this WP is the application of the SEDA scheme to analysing the dialogue. Further, an important feature of the analysis is the deconstruction of talk sequences according to different aspects of computational thinking. Computational thinking concepts such as variables and iteration (*see* 8.3.2.iii) are considered alongside practices such as testing and debugging (*see* 8.3.2.iv) and reusing and remixing (*see* 8.3.2.v).

The WP is divided into two phases, with data collection commencing in phase 2 only. These are now discussed in turn.

8.1.1 Phase 1 – Interactive Tutorials

As this was the culminating WP of the project, a key concern at this stage was transferability. The previous studies showed evidence to support the idea that the microworld-based activities lead to greater improvements in literacy and computational thinking. Further, there was modest evidence to suggest that the activities encourage a more complex, exploratory form of pupil talk. With these positive though limited findings emerging there needed to be consideration given to how other teachers might replicate such interventions effectively within their own pedagogical practice.

When considering the pedagogical problem of introducing programming for the first time, Brusilovsky et. al (1994) outline a framework of three approaches that have been used successfully. First, the *incremental* approach, is where the new programming constructs of a larger language are introduced to the learner gradually over a period of time through a series of programming problems to support learning. Second, the *mini-language* approach is where 'a small and simple language' is designed especially 'to support the first steps in learning programming'. A third *sub-language* approach, though similar to the mini-language approach, is different in that 'while the mini-language approach uses a special miniature language with its own commands and control structures, the sub-language approach provides only a set of commands and queries as an extension of some "big" programming language' (*ibid*., p. 106).

Scratch and Snap!, according to such a framework, are broadly aligned with what Brusilovsky et. al call the 'mini language' approach. Through offering up a series of programming blocks, Scratch positions itself as a smaller, self-contained language to encourage graduation to larger text-based programming environments.

A development of phase one in this WP is a new pedagogical scheme that adopts (i) the staged increase of difficulty and (ii) the problem-based orientation of the incremental approach. Snap! incorporates several software functions that assisted with this. The series of tutorials used in phase 1 of WP5 can be opened in Snap! using the files in the 'phase 1' section of Appendix G.

Addtionally, there is a video providing an example of one of these tutorials. The aim of the tutorial series was to facilitate ease of access to the Poem Generator microworld-based activity.

The tutorials were designed to offer practical exercises on the use of the microworld primitives. The first six weeks of the club involved a small group of learners trying out the primitives to see the effects. Comments were used to provide a narrative of the tasks involved in each of the tutorials. Other blocks were hidden to filter the amount of information on screen at a given time. Learners were required to undertake the series of twelve tutorials, and complete the programming problems contained within them, before progressing to a blank microworld in order to generate their own poetry.

8.1.2 Phase 2 – Tanaga Challenge

Following the initial tutorials, learners subsequently progressed to a blank Snap! microworld with custom blocks for generating poetry. Brusilovsky et al. (1994) explain that a sub-language approach is where the microworld provides 'a set of commands […] as an extension of some "big" programming language' (p. 106). The pedagogy of WP5 begins using an incremental approach with staged tutorials in phase one. Phase two transitions to a sub-language approach. Here learners made use of the extended custom blocks within the larger Snap! programming language.

To manage progression from the tutorials and provide a context to the sessions, it was decided to use a competition brief to structure the six sessions that took place during phase two. This competition brief was called the Tanaga Challenge. The tanaga is a type of short Filipino poem, consisting of four lines of seven syllables each with the same rhyme at the end of each line. The poem uses a 7-7-7-7 syllable form with an AAAA rhyme pattern.

An instruction booklet and a task brief were provided for phase two (*see* Appendix G). The instructions explained the functionality of the microworld according to the typical processes that are required when using it. The key details of the Tanaga Challenge were summarized in a brief distributed to learners.

There was also some development of the microworld activity itself when moving from phase 1 to 2. The number of custom blocks required was simplified in order to make greater use of the original Snap! primitives. Two buttons were also added to overcome the limited space available in the stage area. The first 'show poems' button hides the word boxes in order to view poems on the full stage. A 'hide poems' button was also added to hide the poems when adding items to the word boxes.

Learners were given five sessions to develop their tanaga poems and were asked to present these at the end along with the code that was used to produce them. Such an approach recalls research carried out by Resnick et al. (2005) that advocates design exhibitions when presenting outcomes of programming activities.

In addition to the standard blocks for managing lists that are offered by Snap!, the microworld makes use of the following key custom blocks (*see* Table 22):

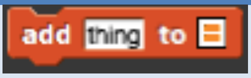
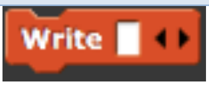
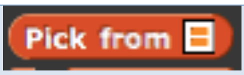
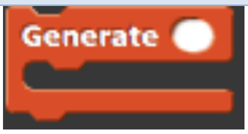
Block	Actions
 add thing to	Used for adding words to the word lists
 Write	Connects plain text and/or word list picks together in order to form a sentence
 Pick from	Inserts a random pick from a word list
 Generate	Repeats the blocks that are in the C-shape for a set number of times.

Table 22: Blocks and their actions

Both the resources and activity files described above are available in the 'phase 2' section of *Appendix G*. There is also a video demonstration of how to make a tanaga using the microworld.

8.2 Methods

This study replicates the pair programming pedagogy (*see* Cockburn and Williams, 2000) and voice recording procedure used in WP4. Recordings were made of several pairs at key points throughout phase two of WP. No data was collected during phase 1 when using the interactive tutorials. Ethical consent was gained from all participants (*see* Appendix C) and all recordings were deleted as soon as they were transcribed. The intervention was led by the teacher-researcher, but an independent academic from the University of South Wales also attended the club for five of the six sessions in phase two. This offers *inter-judge* reliability (*see* Black, 2002) and improves the validity of the data collected.

The key difference with WP5 was that learners were explicitly prompted beforehand to speak aloud (i) the collective actions that were taking place and (ii) the reasons why these decisions were being made. The assumption underpinning this instruction is that the act of verbalizing actions will lead to insights via analysis of a 'dump' of what is taking place in the working memory of learners. This is an assumption that is loosely borrowed from an area of educational research known as protocol analysis (*see* Trickett and Trafton, 2009).

A methodological issue raised by protocol analysis is the extent to which asking participants to 'artificially' verbalize their actions in itself alters the sequence of thoughts that are being spoken. A literature review carried by out Ericsson and Simon (1993), however, found no evidence to suggest that a subject's thought process was changed as a result of the instruction to 'speak aloud' their thinking.

Trickett and Trafton (2009) suggest that such a methodology is particularly effective when working with participants undertaking computer-based tasks. This is because data arising from protocol analysis can play an important part in evaluating the learning that takes place within computer-based environments. As they explain:

> Verbal protocols are likely to be especially useful in evaluation—first, in determining how effective the virtual environment is as a training tool, and second, in assessing the student's learning and performance (*ibid.*, p. 333).

Audio recordings were used in phase 2 and selective screen captures were produced according to how they exemplified concepts and practices in computational thinking. Figure 21, for instance, is one such capture that exemplifies the concept of modularity when thinking computationally, as illustrated by Ethan's custom 'block within a block'. Following transcription of the audio recordings, the pairwork sequences were analysed according to Hennessy et al.'s (2016a) SEDA coding scheme. This coding scheme was selected over Mercer and Wegerif's (1997) coding scheme in order to achieve greater granularity when analyzing dialogic sequences. Whereas WP4 provided a useful broad-grained analysis, SEDA provides thirty-three codes across eight clusters to assign to interactions. Mercer and Wegerif's framework characterizes talk sequences according to three categories of confrontational, cumulative or exploratory talk. As Hennessy et al. (2016a) explain, the benefit of the SEDA scheme is that it enables a 'fine-grained, systematic analysis of what participants actually do and say in practice during dialogic interactions' (p. 4). The SEDA scheme is that it was developed very recently by a collaborative research term who analysed video data derived from contemporary British and Mexican classrooms (*ibid.)*. Further justification for its use, therefore, is that it provides both a timely and context-specific coding frame for analyzing dialogue.

The result of a three-year project in UK and Mexico, the SEDA scheme begins by establishing a hierarchical and nested model for analyzing dialogue. The SEDA scheme identifies communication on a micro, meso and macro level of communicative acts (CA), communicative events (CE) and communicative situations (CS) respectively. A CA is one contribution within a conversation. Several CA about a particular topic or task form a CE. Each turn in the conversation to a different topic signals a change of CE. Finally, the overall general context of the conversation is the CS and this may be comprised of multiple CE.

The analysis used in this WP is fine-grained in that it largely restricts itself to a micro-level analysis of CA only. The CS of the dialogic analysis is the conversation that takes place within the club session with the different excerpts of dialogue being separated into different CE. In order to carry out the micro-level analysis of the different CE, each CA was assigned one of thirty-three codes in eight clusters. It was therefore possible to gain a deeper grain of understanding with respect to the kinds of peer talk exchanges that were taking place as the activities were undertaken. A cluster and code summary of Hennessy et al.'s SEDA scheme is provided in *Table* 23 and is also available online in a condensed (*see* 2016b) and comprehensive (*see* 2016c) form:

Cluster code	Description
I	Invite elaboration or reasoning
R	Make reasoning explicit
P	Positioning and coordination
B	Build on ideas
C	Connect
RD	Reflect on dialogue or activity
E	Express or invite ideas
G	Guide the direction of dialogue or activity

Table 23: SEDA Cluster and Code Summary

8.3 Results and Discussion

The analysis of the dialogue between teachers and learners during the microworld-based teaching scheme begins with a preliminary statistical overview of the clusters for all CA recorded before moving to discuss the twelve CE individually.

8.3.1 Overview

Figure 16 shows the CA cluster frequencies organized by their agents of either teacher or learner. The clusters are deconstructed into individual CA in Appendix G.

It is clear to see that teacher dialogue was considerably less diverse in the scope of CA and clusters in comparison to that shown by learners. Teacher dialogue was largely restricted to either cluster G (where guidance was issued) or cluster I (where reasoning was invited). By making use of careful questioning, teachers spent a lot of the time inviting learners to think carefully about the task in hand and the steps that should be followed next.

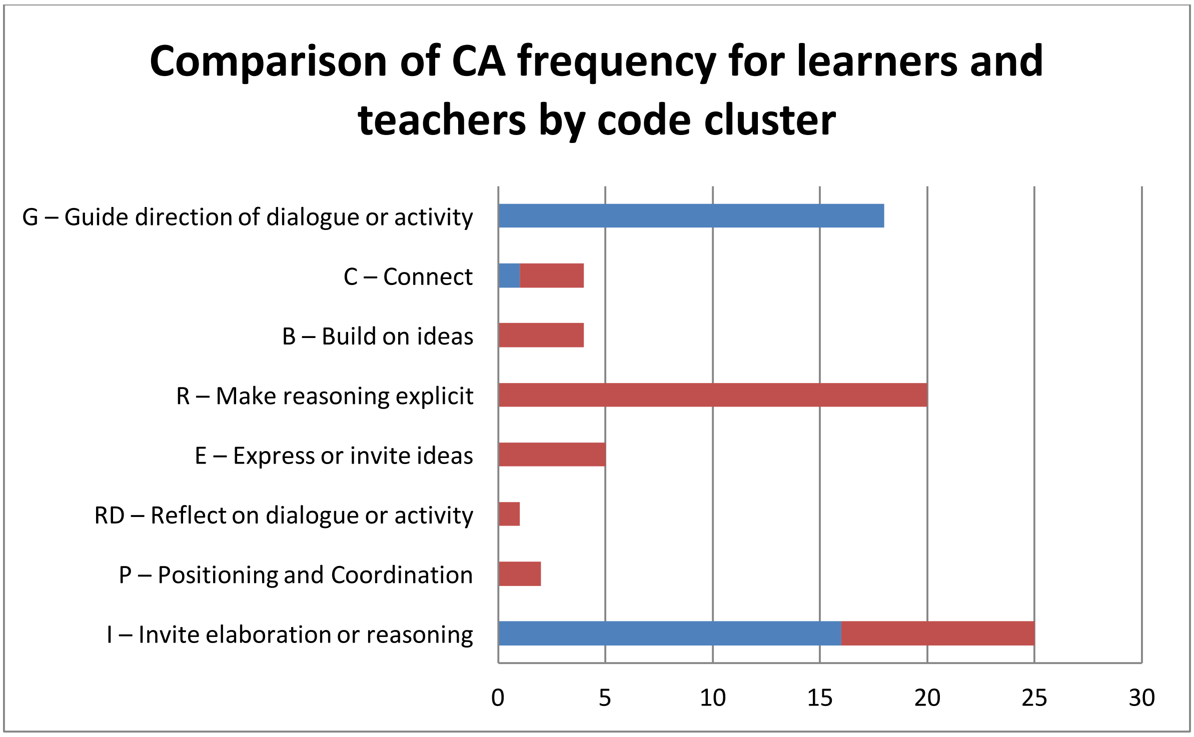


Figure 16: Comparison of CA frequency for learners and teachers by code cluster

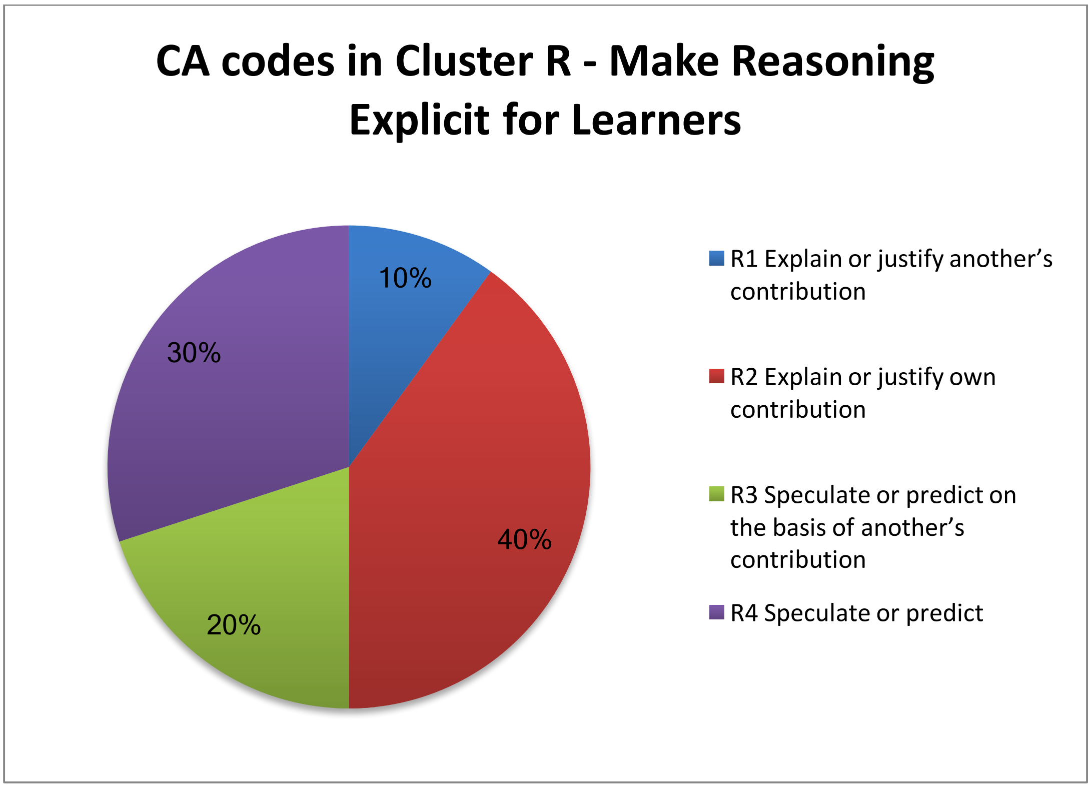


Figure 17: Breakdown of CA Codes in Cluster R – Make Reasoning Explicit for Learners

The range of the interactions between learners, on the other hand, was greater. Still, however, there are two clusters that are particularly stronger than others. First, learners regularly asked their teachers and peers for help in order to take the next steps that were required. This explains the high invitation for elaboration (cluster code I) that is seen here. It is particularly encouraging that the microworld activity, combined with teacher questioning and peer discussion, gave rise to interactions where learners verbally articulated their reasoning or formed hypotheses/predictions (cluster code R). Indeed, this was the largest cluster and accounted for a near majority (45%) of all learner CA that were recorded.

Figure 17 breaks down the CA that were recorded for learners in cluster R. It is particularly pleasing to see that 70% of the recorded CA within this cluster (codes R3 and R4) focus on interactions where learners are verbally establishing theories or exploring possibilities regarding what will happen when a particular action is taken in the microworld. There is evidence to suggest that the microworld activity combines with teacher and peer talk to incite speculation and prediction. This finding is not dissimilar to the exploratory talk that was witnessed using the Wegerif and Mercer (1997) coding scheme in the previous WP. These findings are limited, however, by the relatively small number of communicative events that were recorded (twelve) in addition to the small sample of learners that appear across the CE (five).

8.3.2 Analysis of Individual CE

Following the overview, the remainder of the findings engages individually with the twelve CE that were recorded during phase 2. The twelve CE are grouped according to the context of each interaction and are organized thematically into five loose topics of conversation.

8.3.2.i Lists of Words

The analysis reveals a considerable level of conversation regarding the words to be added to each of the word lists. CE1/2 in *Tables 24* and *25*, shows two conversations that takes place between Ovyaa, Nimra and a teacher regarding the word lists that were available.

CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE1 Ovyaa independently seeks out new words for the word lists	Teacher	1	What are you doing there girls?	I4		
	Ovyaa	2	We need to find different words which rhyme with 'cat' because we didn't like the words in the boxes.	R2	R1	
	Teacher	3	Bat, rat..	G2		
	Ovyaa	4	Yes, there are loads here on this site.	R2	R1	

Table 24: Dialogue CE1

The first CE shows Ovyaa's dissatisfaction with the original rhyming patterns that were available and her desire to create a new rhyming pattern '-at'. Ovyaa decided she wanted '-at' rhyming words and so carried out a keyword search to locate an online word list. Though this appears a modest step, it was pleasing to see the creativity demonstrated here as Ovyaa rejected the many rhyming word lists that she was provided with in favour of following her own approach. Rather than spending time merely thinking up her own words, Ovyaa carried out an online search in order to find inspiration.

The second CE shows Nimra making an important discovery about the nature of the microworld and how it was constructed. Through careful structuring by the teacher, Nimra makes the correct hypothesis in line 3 about how to clear the list of poems. She works out that a 'delete' block can serve to 'reset' the display in the same way that the same block can be used to 'empty' the word lists. The link that is made following the intervention of the teacher is important since it says something about the commonality of 'lists' as a data structure in programming.

CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE2 Nimra makes a discovery regarding the Poem Generator	Nimra	1	Sir, how would I delete all the poems on there.	I4		
	Teacher	2	Look again at the blocks list. Remember that the Poem Generator is just another word list – another box of words. A variable.	G2		
	Nimra	3	Would you use the delete like with the word boxes?	R3	C1	
	Teacher	4	Yes, you've got it.	U		

Table 25: Dialogue CE2

8.3.2.ii Rhyming Patterns

The second theme emerging from the conversational data is the focus on different patterns of rhyme. CE3 (*Table 26*) shows a conversation that takes places between Ethan, Nimra, Ovyaa, and the teacher regarding the creation of a new word list. This dialogue serves as a useful example of how the microworld is able to facilitate shared knowledge construction and peer learning. Ovyaa, learning through both Ethan and the teacher's direction, decides to create a new word list with an alternative rhyming pattern in order to enhance the poetry that she generates.

Table 27 in CE4 illustrates the extensibility or 'high ceiling' of the microworld as Ethan is set the extension task to use what he has learned to invent his own poetry form. At line 3, the teacher introduces Ethan to a limerick found online and Ethan responds to teacher-directed questioning in order to arrive at the correct syllabic and rhyming structure for a limerick in line 7. Undertaking the challenge, Ethan learned about the syllabic structure of a particular poem type: a tanaga. It may be suggested from this CE that Ethan, with careful teacher scaffolding, has taken this knowledge and applied it to a different problem in order to recognize a different poetry form.

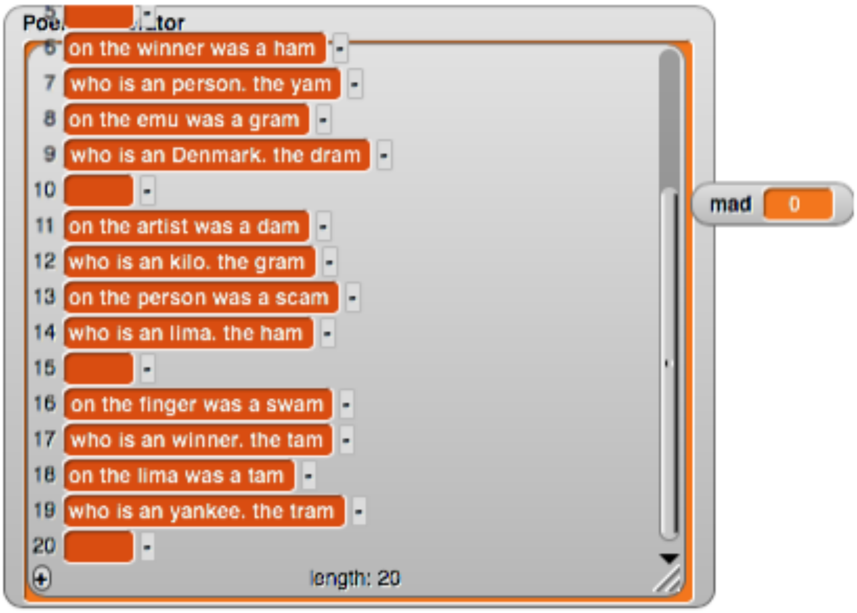
CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE3 A whole class discussion regarding rhyming patterns	Teacher	1	What rhyming pattern is that, Ethan?	I4		
	Ethan	2	The first two are the same and then the next two are different.	R2		
	Teacher	3	So, girls, how would you change the rhyming pattern like Ethan?	I4		
	Ovyaa	4	Add a new box for the different pattern.	R3		
	Teacher	5	Good.	U		

Table 26: Dialogue CE3

CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE4 Ethan experiments with alternative rhyming patterns of his own accord and is then given an extension task by the teacher	Teacher	1	What have you got there, Ethan?	I4		
	Ethan	2	There are two different rhymes at the end of my poems.	R2		
	Teacher	3	That's great. It is not a tanaga but we would say that has an A-A-B-B rhyming pattern. Have a look at this *(starts a keyword search for 'limerick' in a new window).*	G2	G4	
	Teacher	4	How many syllables are there?	I4		
	Ethan	5	Ten. Ten. Six. Six. Ten.	R3		
	Teacher	6	What do you notice about the rhyming pattern of this poem?	G6	G2	
	Ethan	7	Those two lines (*pointing at lines three and four*) are different rhymes.	R3		
	Teacher	8	Good. Can you make your own limerick?	G2		
	Ethan	9	I'll try!	U		

Table 27: Dialogue CE4

Due to time constraints, Ethan did not progress to create a limerick by the end of the tanaga challenge, though he did successfully produce a series of tanaga poems including a custom block for future use. The custom block he created is discussed in the next CE. *Figure 20* shows a screen grab of the tanaga poems he created using the microworld.



Poe... ...tor
6 on the winner was a ham
7 who is an person. the yam
8 on the emu was a gram
9 who is an Denmark. the dram
10
11 on the artist was a dam
12 who is an kilo. the gram
13 on the person was a scam
14 who is an lima. the ham
15
16 on the finger was a swam
17 who is an winner. the tam
18 on the lima was a tam
19 who is an yankee. the tram
20
length: 20

Figure 20: Ethan's tanaga poems

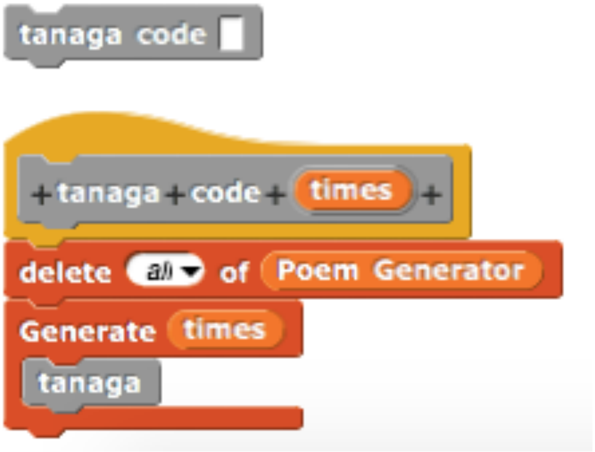
8.3.2.iii Creating Custom Blocks

Third, the creation of custom blocks featured as a key theme within the class discussions. CE5 (*Table 28*) is an account of conversation between Ethan and the teacher as Ethan creates his own custom block for generating a tanaga poem. Ethan initiates the conversation by seeking teacher approval of the custom tanaga block that he creates.

CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE5 Ethan creates a custom block that generates a tanaga for a set number of times	Ethan	1	Sir! Look, (*pointing at the screen and a custom 'tanaga' block*) I've made a tanaga with only four blocks.	B2		
	Teacher	2	Great. Now can you also create a custom block containing the 'delete' and 'generate' blocks?	G2		
	Ethan	3	Ok. How do I make a block with a number space in for the number of poems?	R4	I4	
	Teacher	4	Good idea - you mean a block variable? You create a space like this *(demonstrates)* and then add this into the generate block, like this *(demonstrates)*.	G2		
	Teacher	5	Can you make your own and test if it works?	G2		
	Teacher	6	Yes, thanks!	U		

Table 28: Dialogue CE5

A key issue with the custom block Ethan made, however, is that there is no method of incorporating bounded iteration to generate each poem a set number of times. In line 3, Ethan asks a very important question and explains (though not using the correct terminology) that a block variable is required in order to generate a series of poems. A block variable is a variable that exists locally to a block, effectively enabling the set of instructions inside the block to be looped a set number of times. The sequence ends with the teacher explaining how to do this and Ethan making his new block to solve the problem. The resulting custom block that Ethan produced, and the contents of this block, is shown in *Figure 21.*



```
tanaga code [ ]

+tanaga+code+ (times) +
delete (all ▾) of (Poem Generator)
Generate (times)
    tanaga
```

Figure 21: Ethan's custom block

CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE6 During his presentation at the end of the challenge, Ethan explains the new block he has created.	Teacher	1	What block have you got there, Ethan?	I4		
	Ethan	2	You type in the number of times you want a tanaga in the space in the block.	R2		
	Teacher	3	What is inside that block?	I4		
	Ethan	4	There is another tanaga block that makes the four lines.	R2		
	Teacher	5	Show me.	G2		
	Ethan	6	(*Demonstrating*) Here is a block within a block.	E2		

Table 29: Dialogue CE6

CE6 in *Table 29* takes place during the showcase at the end of the tanaga challenge. In line 3, the teacher asks Ethan to explain the construction of the block in *Figure 21*. Towards the end of the conversation, in line 5, Ethan makes a very important discovery about computation. Ethan realizes that he has created a custom programming block that incorporates a different custom block he has also created. Ethan created a custom block called 'tanaga'. His second block, unusually named ‘tanaga code’, contains the tanaga block with the addition of a block variable to specify the number of times the generator will repeat. This block variable enables the user to define the number of the times a tanaga is generated. Through making his custom block within a block, Ethan has made a discovery about the modular and hierarchical nature of programming code.

CE7 in *Table 30* illustrates the ‘block within a block’ concept again, this time in conversation with Nimra. It is particularly pleasing to see here that the modular nature of programming appears to affect Nimra’s view of how poetry forms are constructed. Nimra begins by, in line 1, asking the teacher for assistance to create a custom block. In line eleven Nimra discovers that, in order to create a tanka block, a haiku block can be used with only two additional lines. Key to this discovery, however, is the connection that Nimra makes between the haiku and tanaga poetry forms. Nimra realizes that a tanka contains a haiku within the first three lines of its syllabic structure. Nimra is making a comparison about the construction of two different poetry forms as a result of her programming experiences.

CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE7 Discussing the process of constructing a programming block within a programming block for haiku and tanka forms	Nimra	1	Sir, how do I put this *(gesturing towards the haiku blocks in the scripting pane)* in a block?	I4		
	Teacher	2	First you click this *(pointing towards the ‘make block’ button)* and then you drag in the blocks you want.	G2		
	Nimra	3	*(Creates new block)* Ok, so where would I go for it?	I4		
	Teacher	4	Where are all of the other blocks?	C1	I4	
	Nimra	5	Here - *(Gesturing to the screen)* Oh, here it is!	B2		
	Teacher	6	Good. Now how would you make a tanka?	I4		
	Nimra	7	Is a tanka 5-7-5-5-5 syllables?	I4		
	Teacher	8	Yes.	U		
	Nimra	9	Would you have an haiku block and then add another two lines of 5?	E2		
	Teacher	10	You could-	U		
	Nimra	11	Ohhh… Could you then put an haiku block and the new two in a new block?	R4	E2	
	Teacher	12	Yes.	U		

Table 30: Dialogue CE7

8.3.2.iv Debugging

A prevalent theme within class discussions was the debugging process within programming. These conversations focus on a key practice of computational thinking identified by Resnick and Brennan (2012). CE8-10 (*see Tables 31 – 33*) provides three examples.

The first of these, CE8 (*Table 31)*, begins with Ethan's frustration that the poem generator stops each time it reaches a particular line of the poem. Guided by careful teacher-directed questioning, Ethan discovers in line four the reason for this error. He has typed an input into the random pick block that references a word list that does not exist. The CE ends with Ethan removing the block and testing that the change works. The questioning by the teacher in this conversation is particularly open at lines two and four: the problem solving skill demonstrated by Ethan here in locating and resolving this problem is particularly impressive.

CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE8	Ethan	1	Sir, why is this not working?	I4		
Debugging an issue that prevented the 'generate' block from functioning correctly	Teacher	2	Have a look at your blocks – is anything stopping it?	G2	I4	
	Ethan	3	Oh.. Is it because of 'bad' (a custom word list)?	R4		
	Teacher	4	Why would that make a difference?	I4		
	Ethan	5	Its empty. No words in it.	R4	B2	
	Teacher	6	Try taking it out then.	G2		
	Ethan	7	It works now, thanks!	U		

Table 31: Dialogue CE8

CE9 (*Table 32*) is a conversation between Ovyaa and a teacher. At first in line one, Ovyaa wonders why the words generated have been displayed without spaces and all at once. By line five, however, Ovyaa makes the realization that the words she is seeing is a complete list of all items within the word list. Ovyaa realizes that she has not added a pick random block in order to choose one particular item from the list. As such, the entire word list has appeared in the poem generator on the stage. This example not only illustrates Ovyaa's debugging capability: she has also discovered an important programming concept that will prove useful later in the education cycle. This concept is the difference between lists and the data items that are contained within them.

Finally on debugging, CE10 provides a useful sequence in *Table 33*. This sequence is not dissimilar to the other debugging issues. Here, however, the problem is slightly different as a random pick block is included but without an input to specify the word list to pick from. This time more detailed teacher questioning is needed in order to elicit a helpful response. This is a useful sequence, however, for a different reason. Papert (2008) uses the phrase 'hard fun' to articulate the perseverance and enjoyment of undertaking programming activities. This feeling, perhaps, is illustrated particularly well in Rokas' exclamation that 'this hurts my brain' in line two.

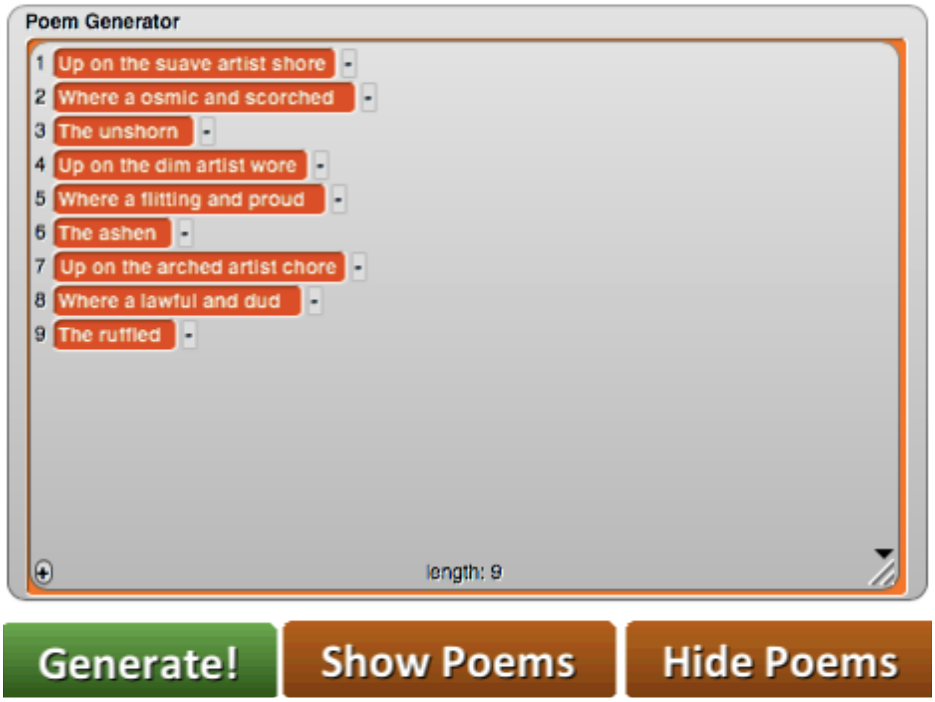
CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE9 Ovyaa uses debugging to correct her mistake and realizes the computer's confusion with handling lists	Ovyaa	1	Sir, its not working. There's no spaces anywhere on here.	I4		
	Teacher	2	Which blocks have you used?	G6	I4	
	Ovyaa	3	Oh, there is no 'pick random' on there (*laughs*)!	R4		
	Teacher	4	So where have those words come from then?	I4		
	Ovyaa	5	It has returned all of the words rather than just one of them.	R2		

Table 32: Dialogue CE9

Rokas did not attend all of the extracurricular sessions and therefore did not complete the tanaga challenge successfully. He was able to generate some lines of poetry, however, which included random picks from the word lists. Figure 22 shows a screen grab of his work.

CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE10 Rokas hypothesizes about picking from different lists and then debugs an error	Rokas	1	(*Reading aloud to self*) Upon the dim artist moor where a pavid and – noun – scores things. Score things? That makes no sense!	U		
	Rokas	2	(*To self*) I'm going to try – lets try – okay – What? No, not verb. I don't want verbs. Okay. I'm going to try nouns. This hurts my brain.	R4		
	Rokas	3	Sir, why is there an error? There is an error!	I4		
	Teacher	4	Press 'Generate'	G2		
	Rokas	5	(*Reading error*) The list length is not a function.	E2		
	Teacher	6	I see why. Where are you trying to pick from? (*Pointing towards an empty item block).*	G6	G2	
	Rokas	7	Oh!	U		

Table 33: Dialogue CE10



```
1 Up on the suave artist shore
2 Where a osmic and scorched
3 The unshorn
4 Up on the dim artist wore
5 Where a flitting and proud
6 The ashen
7 Up on the arched artist chore
8 Where a lawful and dud
9 The ruffled
length: 9
```

Figure 22: Rokas' Lines of Poetry

8.3.2.v Reflecting on Learning to Program

The final theme revealed by the dialogue analysis were interactions relating to the process of learning to program. CE11 (*see Table 34*) is a conversation between students James and Rokas. James was a newcomer to the club and the conversation focuses on Rokas showing James how to use the microworld. On line four, James panics 'I broke it' as he struggles with the process of making a random pick from a word list. In line five, however, Rokas reassures James by advising him to 'copy' his (Rokas') code at first before expanding his own program later. The interaction between Rokas and James exemplifies the computational thinking practice of reusing and remixing. As documented in table 34, James begins by copying Rokas' code (reusing), and Rokas advises James that he (James) later needs to make changes to the copied code in order to create something new (remixing).

CE12 (*see Table 35*) is a conversation between Ethan and a teacher. The conversation begins with Ethan showing the teacher a block he created to make a tanaga. As a result of questioning from the teacher at line six, Ethan responds in line seven by speaking about his desire to keep the program 'neat' and 'all in one block'. Ethan reflects here on the 'quality' of the code that has been produced and provides a rationale for why a custom block is needed. This presents an impressive and unprompted realization that code can be 'good quality' or 'poor quality'.

CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE11 Rokas introduces newcomer James to the club and attempts to scaffold James' learning	Rokas	1	This is an haiku (gesturing towards James' screen). This is going to be one, two, three, four. Four syllables. You need seven in this line. You need to figure out a way of adding seven to this line.	P5		
	James	2	Item random. How do I get item random?	I4		
	Rokas	3	You find it here (pointing towards the item block). You then get one of these boxes and put them in there.	E2		
	James	4	I broke it.	U		
	Rokas	5	James (reassuringly).. don't touch anything. Just try copying mine to start with.. don't worry James.	P3		

Table 34: Dialogue CE11

CE	Agent	Line	Discussing the Poem Generator microworld	CA		
				Code one	Code two	Code three
CE12 Ethan reflects on the wider importance of thinking computationally	Ethan	1	Look, Sir, I've made a tanaga.. I remember you showing us to make a block.	R2	C1	
	Teacher	2	Great. What have you put in there?	I4		
	Ethan	3	Four write blocks and an –ash.	B2		
	Teacher	4	Ash? The rhyming pattern?	I6		
	Ethan	5	Yes.	U		
	Teacher	6	Why did you decide to do that?	I4		
	Ethan	7	Because if your boss came over he would want to see it all in one block and made looking neat.	RD2	C3	

Table 35: Dialogue CE12

8.4 Conclusions and Limitations

From the analysis of teacher-learner and learner-learner dialogues some clear patterns emerge.

Teacher talk focused on situations where guidance or instruction was issued to learners; where they suggested the next steps required in order to make progress. This finding matches with expectations. Teacher dialogue also prompted learners to use reasoning to solve problems in the microworld. During the activities, teachers used questioning characterized by key words such as 'how', 'why' and 'what would happen if?' in order to encourage reasoning.

A more surprising finding from the cluster analysis is the comparatively larger range of interactions that were witnessed when analysing the learner responses. Learners built upon the ideas of others in a way that is not dissimilar to the cumulative talk sequences identified in Wegerif and Mercer's (1997) framework. There were also similar invitations to elaborate reasoning through questioning that were witnessed in the teacher interactions and evidence to suggest that the activity incites reflection upon the value of what is being learned (see, for instance, CE12).

By far the most frequent cluster code amongst learners, however, was cluster R where reasoning was made explicit. Many of the CA within this cluster were derived from situations where learners explained or justified their thoughts through answering the questions posed by teachers and peers. Within the reasoning cluster, half of the CA recorded by learners were characterized by prediction, hypothesizing or exploration. The individual analyses of the twelve CE provide insights into the ways in which talk inferring reasoning manifested itself. Another way of expressing this data is to examine the profiles of some key learners.

Nimra and Ovyaa were regular attendees at the club. Both learners are high-attaining and they appeared to care a great deal about the creative/aesthetic aspect of the poems that they created. The girls initially rejected the set word lists of rhyming patterns that they were provided with. Following teacher guidance, the girls conducted a keyword search on the Internet for examples of words that fit a more unique rhyming pattern. Their investigation into finding a unique word pattern led to them creating a new word list.

Rokas attended the majority of Code Club sessions and was joined on one occasion by his friend James. When his friend James joined, it was interesting to see Rokas reflect on the pedagogy of learning programming by suggesting that his friend copies – or remixes – code to begin with before progressing to adapt this code for his own purposes. Rokas was particularly good at debugging and an example of this is where he solved the problem of trying use a pick block without specifying a word list (see CE8).

The winner of the tanaga challenge was Ethan. His poems met the brief (*see* figure 54) and were accomplished using a high level of programming skills. Throughout the challenge, Ethan pushed the boundaries of what was expected of him. He created a custom block to create a tanaga but his own inquiry led to him create a block variable. The way in which he incorporated bounded iteration within his block design is impressive and he quickly produced a tanaga that he was satisfied with. More pleasing was how Ethan was able to take the literacy knowledge regarding poetry form and apply this to the extension task of deconstructing a limerick with different rhyming pattern with teacher guidance.

The usefulness of the classroom talk analysis is that it shows the many different forms of exploration that are enabled through the *guided discovery* orientation of the poem generator microworld. Nimra and Ovyaa improved their cross-curricular literacy skills by sourcing and providing examples of vocabulary with different rhyming patterns. Ethan developed his computational thinking skills in abstraction and looping by constructing his own block that incorporated bounded iteration. Rokas, on the other hand, makes what could be termed a *mathetic* (1993a) discovery about how learning happens when programming. In 1991, Turkle and Papert (1991) use the term *epistemological pluralism* to describe multiple ways of knowing. The success of the microworld activity used in the club, perhaps, is that it provides a constructionist learning environment that allows for making many different but purposive discoveries.

The analysis also revealed the importance of dialogic teaching as part of the process of learning with microworlds. *Monologic* classroom talk is where the aims of classroom interactions are controlled by the teacher: ‘communication geared towards achieving the teacher’s goals’ (Lyle, 2008, p. 225). *Dialogic* teaching, on the other hand, ‘creates a space for multiple voices and discourses that challenge the asymmetrical power relations constructed by monologic practices’ (*ibid.*). Alexander (2006) identifies dialogic talk by some defining features. First, dialogic talk is *reciprocal*: the teacher listens to the learner’s questions and ideas *in addition* to posing questions themselves. Second, dialogic talk is *cumulative*: teacher responses build upon issues posed by the learners *as well as* the other way around. He suggests that the movement towards dialogic talk in the classroom sees the learner taking more control over asking questions and commenting on ideas as he/she encounters opportunities for creating knowledge in the classroom.

> Dialogic teaching […] explores the learner’s thought processes. It treats students’ contributions, and especially their answers to teacher’s questions, as stages in an ongoing cognitive quest […] it nurtures the student’s engagement, confidence, independence and responsibility (*ibid.*, p. 35).

There are numerous examples in the CE data where questioning was initiated by the learner as a result of the issues that they encountered when exploring with the microworld. There is evidence to suggest that learners constructed new knowledge through the process of talking about these issues with either a teacher or a fellow learner. In CE5, for instance, Ethan carefully frames a question to the teacher regarding how to incorporate looping into his block design. This led to the successful development of a new custom block that incorporated bounded iteration. In CE7, an exchange between Nimra and the teacher (that was initiated by Nimra) led to the discovery that within every tanka is a haiku. These examples provide evidence to suggest that dialogue is particularly important to the process of constructing new knowledge when exploring within a microworld. When following a microworlds-based approach, therefore, dialogic teaching should figure as a pedagogical recommendation.

9 Conclusions

LOGO, used in the microworld projects in the 1980s, is an extensible programming language that enables new primitives to be created. This project has presented an examination of whether block-based programming languages, allowing for custom block creation, may be used in a similar way by educators in order to develop specific, deconstructed aspects of literacy and computational thinking. The project provides justification for using a block-based approach through the learners' development of a diverse range of computational thinking concepts and practices (*see* section 9.2.1 for a summary).

The concluding remarks of the project begins with a review of its limitations before moving to consider its key findings. Finally, the thesis ends by identifying how this research may impact upon classroom practice and how this fits with current education policy developments in Wales.

9.1 Limitations

This research project has demonstrated, through multiple studies that took place in two South Wales secondary schools, that there is evidence to support the impact of a microworlds-based pedagogy upon aspects of literacy and computational thinking development. A key strength of the project is the action research approach which, combined with the work package structure, enabled a cycle of planning and evaluation to inform the design of the individual studies. A further strength is the mixed methods used across the work packages in order to triangulate findings. It is appropriate at this stage, however, to discuss the constraints on the interpretation of the findings.

9.1.1 Sampling Limitations

All but one of studies carried out here take their samples from a single school population. WP2 took place at a different secondary school (School 2) and local authority. An analysis of school contextual data (Estyn, 2011 and Estyn, 2016) reveals that eFSM (free school meal eligibility) continues to remain well below the national comparator at School 1. More research is needed, therefore, in schools where learners come from backgrounds of greater socioeconomic disadvantage.

In addition the sample sizes for each of the studies were relatively small with typical intervention and comparison group sizes of less than thirty learners. Each of the classes were mixed-attaining (WP2 is an exception) and mixed gender but these were arrived at using opportunity sampling rather than random selection. Participants were selected according to three considerations: (i) learner consent; (ii) consent of the parent/guardian and (iii) school timetable constraints.

Such limitations, however, are necessary as a practical consequence of an action research methodology. Action research is well established as a paradigm of research in education and can be effective at improving teaching practice at a local, institutional level (*see* Denscombe, 2002; Cohen, Manion and Morrison, 2011). This method is appropriate given that the project aims to bring about positive pedagogical change within the school where the teacher-researcher is employed.

9.1.2 Methodological Limitations

Despite these benefits of action research, there are further limitations that must considered when evaluating the strength of the findings. A quasi-experimental design with a comparison group was used for the quantitative element because a true experimental design with a control group was not possible.

A potential criticism that may be targeted at the quantitative aspect of the research is the extent to which the pre-test and post-test scores are representative of the aspects of literacy and computational thinking that were being assessed. It may be argued that the results simply show that learners in the intervention group improved to a greater extent than their comparison group counterparts when taking the style of tests that were used. One could question, in other words, whether the test designs used were the most appropriate instruments for assessing underlying concepts in literacy and computational thinking.

To address these possible criticisms, great care was taken throughout the WPs to ensure that the test designs mapped appropriately to aspects of computational thinking (Resnick and Brennan, 2012) and literacy (DfES, 2013) within recognised frameworks. Further, the use of qualitative instruments triangulates the quantitative data that was analysed.

Finally, the issue of familiarity (*see* Bell, 1993) and the associated issue of bias is particularly important in action research studies such as this one where the researcher is fully participant within the organization. The majority of the work reported here took place at a school where the teacher-researcher is employed as a full time teacher of ICT. The partner teachers in the curriculum area of English, and English and Drama in the later WPs, both volunteered to participate as a result of a keen interest in technology enhanced learning. It could be argued, therefore, that the positive findings reported here may attributed to investigator bias.

To minimize the impact of familiarity, two academics from the University of South Wales (an organization independent from the school where the research took place) attended the WP3 intervention for one hour. In WP5, an academic independent from the University of South Wales attended the intervention for a total of five hours. These functioned as mechanisms of inter-judge reliability (*see* Black, 2002) within these studies.

9.2 Key Findings

Subsequent to WP1, this project has been guided by three questions across the remaining WPs. Each of these questions will now be discussed in turn with the key findings cross-referenced where appropriate.

9.2.1 GQ1. What is the potential of a microworld-based teaching approach for the development of specific aspects of computational thinking?

Following the initial trial of microworld activities in WP1, self-reporting data were gathered in the form of digital learning journals. The learning journals were completed by learners using a simple online Moodle database activity. Learners were presented with five computational concepts and asked whether they felt they improved in each concept identified. The data revealed that 97% of learners reported an improvement in their computational thinking skills (*see* section 4.3). This was corroborated by data collected in WP3 where a majority (81.82%) reported an improvement following the intervention in computational thinking concepts such as sequencing. Self-reporting data, though useful, is subject to validity problems and is triangulated through the use of pre-tests and post-tests in WP3.

An interesting discovery emerges when the self-reporting data from WP1 and WP3 are decomposed into individual computational concepts. Both sets of self-reporting data score *sequencing* as the computational thinking concept that is most developed by undertaking the microworld activities. This finding is particularly important because it is corroborated by the quantitative findings in WP4. Both sources of data suggest that BYOB and Snap! are particularly useful in developing the application of sequence in computer programs.

Sequencing, defined by Brennan and Resnick (*ibid.*) as the ability to put together programming instructions to create a desired behaviour was particularly effectively developed. WP4 used separate independent samples t-tests to identify gains in computational concepts and practices (*see* Brennan and Resnick, 2015) that were mapped to questions in the pre-tests and post-tests. The gain reported for the sequencing question type achieved the highest confidence level of 92% (p = 0.079) of all question types (*see* section 7.3.1.iii).

The justification for using block-based programming, however, moves beyond the gains that were seen with the computational concept of sequencing. Papert (1980d) reminds us that learning is most effective when the task being undertaken 'goes together with' (p. 202) with the prior experiences of the learner. The term *syntonicity* can be used here to describe how programming with a block-based language recalls learner experiences of assembling real-life LEGO bricks. The evidence demonstrates that block-based programming can lead to gains in many other aspects of computational thinking. In CE5, for instance, Ethan incorporates bounded iteration into the design of a custom block. In CE7, Nimra shows multiple levels of abstraction by placing a custom block she creates within a different custom block. In CE9, Ovyaa engages with processes of testing and debugging whilst developing her poems. The value of the block-based approach to computational thinking development, then, extends to a multiplicity of concepts and practices.

Computational thinking development was also measured as a whole (without deconstructing the question types into specific computational thinking concepts and practices). T-tests were used to measure confidence levels and Cohen's D calculations for effect size in WP2 (*see* section 5.3.2.i), WP3 (*see* section 6.3.1.ii) and WP4 (*see* section 7.3.1.i). Whilst there were statistically significant improvements in the latter two of these WPs, the result in WP2 produced only an 80% confidence level. Effect size calculations align with this trend with a moderate impact on computational thinking gain in the latter studies but only a modest-to-moderate effect in WP2. WP2 was carried out at a different school and consisted of two lower-attaining groups that were streamed by academic attainment level. WP3 and WP4, on the other hand, were carried out in mixed-attaining classes that were not streamed. The results show that the microworld intervention had a greater effect in the mixed-attaining settings. Since the lower-attaining group featured in an earlier phase of the project (WP2), it is possible that may be due to a lack of refinement in the microworld design. More data is needed that focuses on interventions taking place with classes that are streamed according to academic attainment.

A further finding to guide future work is the link between literacy and computational thinking performance. WP2 shows a modest-to-moderate ($r = 0.341$) positive correlation between improvements made in the literacy tests and improvements made in the computational thinking tests with a confidence level greater than 90% ($p = 0.082$) (*see* section 5.3.3). The regression model in WP3 also provided evidence to corroborate this finding (Beta = 0.228, $p = 0.136$) when computational thinking performance was used as a predictor of performance in literacy, although the confidence level was slightly weaker (*see* section 6.3.1.iii). These findings are interesting and are supported by work such as that by Howland and Good (2015), which draw similar parallels between the written word and coding. This finding also links to Sharples (1985) and the idea of a dual sense of debugging in terms of both linguistic, and computational, models.

The peer talk analysis that was undertaken in WP5 provides qualitative evidence to corroborate further the quantitative findings in the earlier WPs. There are several examples of where learner dialogue demonstrates an engagement with computational thinking concepts and practices. CE2 shows Nimra making a discovery about lists as a common data structure in programming (*see* section 8.3.2.i). In CE6, Ethan engages with the modularity of programming by making a custom block within another custom block, complete with a local variable for bounded iteration (*see* section 8.3.2.iii). And in CE9, Ovyaa debugs her program and correctly identifies where she made an error in a programming sequence (*see* section 8.3.2.iv).

In summary, there is both quantitative and qualitative evidence that supports the potential of the microworld-based approach for developing computational thinking. There is evidence here of a link between performance in the aspects of literacy and computational thinking that were tested for. The quantitative improvements seen were more marked in mixed-attaining classes as opposed to a lower-attaining group that was streamed. This provides an implication for future development of this research.

9.2.2 GQ2. What is the potential of a microworld-based teaching approach for the development of specific aspects of literacy?

The microworld and test designs that were used across the studies focused on specific aspects of the Welsh Government's Literacy Framework (DfES, 2013). The expectation statements targeted were year 7 (eleven-to-twelve year old) literacy skills in adapting grammar, sentence and whole-text structures. The quantitative study that took place with low-attaining learners (WP2) revealed that the microworld had a modest effect (d = 0.46) on literacy skills with close to an 80% confidence level for change (p = 0.242) (*see* section 5.3.2.i). In a mixed-attaining setting, WP3 revealed a statistically significant impact on the literacy aspects (p = .029) and a moderate effect size (d = 0.63) (*see* section 6.3.1.i). WP4 again saw a moderate effect (d=0.54) with a 90% (p=0.092) confidence level for change (*see* section 7.3.1.i). As with the computational thinking t-tests, this evidence shows that the microworld activities had a more powerful impact in the literacy aspects that were tested for when used within a mixed-attaining setting.

The PLaNS (Playful Learning and Narrative Skills) project at the University of Cambridge (*see* Whitebread and Basilio, 2015) has recently used pre-tests and post-tests to measure improvements in narrative skills when primary age learners participated in a guided play intervention. Learners first planned their stories using physical, real-world LEGO construction blocks before progressing to express their ideas in written form. The playful intervention led to a significant improvement in the quality of writing demonstrated.

The finding by the PLaNS project (*see ibid*.) aligns with a related finding here. In WP4, it was possible to map the different expectation statements in literacy to the individual test questions. When learners were asked to write their own poem using a different fixed syllabic form, a statistically significant (p = 0.047) gain was demonstrated (*see* section 7.3.1.ii). The microworld-based intervention using the virtual Snap! blocks aligns with a greater improvement in the ability to structure fixed-form poetry. It is interesting to note that this aspect of attainment data in WP4 is further corroborated by the self-reporting data in WP3 where 90% of learners thought that their understanding of different fixed-syllable poetry forms had improved (*see* section 6.3.2.i).

A second parallel can be drawn between both the PLaNS and this microworlds project. Whereas this project saw more pronounced gains when the interventions took place with the mixed-attaining groups, the PLaNS project also reported that 'the children who were more playful and creative […] were […] those that worked better in their *mixed ability* [*emphasis added*] group (*ibid.,* p.15).

The self-reporting data also revealed that only 61.9% of learners thought their ability to recognise different word classes had improved. This finding contradicts the attainment data in WP4 however, where a good gain was reported with a confidence level of around 85% (p = 0.153) (*see* section 7.3.1.ii).

Qualitative data derived from peer talk analysis further contributed to this research question. Wegerif and Mercer's (1997) framework for peer talk in WP4 suggests that the microworld activity prompted a dialogic exchange that was exploratory in type and focused on a particular aspect of literacy. We see this in the sequence by Scarlet and Morgan (*see* section 7.3.2.iii). Here, Morgan's critical challenge in this sequence appears to lead to a co-construction of knowledge about different word classes. The discourse analysis in the final WP uses the more comprehensive SEDA (Hennessy et al., 2016a) coding scheme and provides further evidence of reasoning with respect to literacy. CE7, for example, shows Nimra positing a link between the haiku and tanka poetry forms that follows on from the action of making a custom block (*see* section 8.3.2.iii). Further, CE4 shows Ethan deconstructing the rhyming and syllabic pattern of limerick (*see* section 8.3.2.ii).

There is both quantitative and qualitative data, then, to support the idea that a microworlds-based approach has the potential to support literacy development. The evidence further identifies some particular aspects of literacy that may be developed using this approach: the structure of fixed form poetic writing is one of these areas. It is interesting to note that the computational thinking concept of sequencing was also particularly well developed. This indicates a link between the ability to sequence a set of instructions in a computer program and the ability to structure poetry and prose, though this idea requires further coverage.

9.2.3 GQ3. What recommendations can be made for microworld-based classroom practice?

The self-reporting data on the pilot of activities for many different kinds of subject domain (WP1) helped to identify the curriculum focus on literacy that figured as a key question to guide the remainder of the thesis (*see* section 4.3). Whilst creative applications of programming figured prominently in the theoretical positions consulted in the literature review, the self-reporting data revealed that only a minority of learners felt the microworlds approach was effective in the curriculum areas of music and art and design. A larger number of students, however, felt that the application of a microworlds-based pedagogy to the curriculum area of English was more effective. It is important to state the caveat, however, that this finding may be more reflective of the design quality of the microworld activity as opposed an intrinsic suitability of this domain to the microworld-based approach.

From this research, a number of recommendations may be outlined that will support the development of microworld-based learning approaches. Recommendations one to seven are broadly focused on pedagogical design, though recommendations eight and nine should not be ignored due to the necessity for sustained professional support and models of curriculum change.

R1. Language learning is a constructive context for microworld-based learning

This research project has contributed new data, derived from a realistic classroom-based setting, to support literacy development as a context for learning with microworlds. WP4 quantitative data revealed aspects of language development that are particularly susceptible to microworld-based approaches to learning (*see* section 7.3.1.ii). Following the microworld intervention, a statistically-significant improvement was demonstrated in the ability of learners to adapt their writing structures to fit fixed verse forms. The improvement in learners recognizing different word classes was also strong following the intervention; achieving a confidence level of around 85% ($p = 0.153$). These two particular aspects of literacy, it follows, are recommended foci when constructing further microworld activities for language development.

The claim that programming is effective for exploring language already has a theoretical basis developed both at the time of the original microworlds research (e.g. Sharples, 1985) and more recently with Scratch (Burke and Kafai, 2010). Howland and Good (2015) also suggest such a link between literacy and programming in their creation of the new programming tool *Flip*. *Flip* places block-based programming alongside natural language in the creation of 3D role-playing games.

An evaluation study with pre-tests and post-tests involving a small number of learners reported significant improvements in communication about computational thinking concepts following use of Flip.

R2. A carefully designed *teaching scheme*, with appropriate resources, should increase access to the microworld

As the research progressed, it became evident that more detailed consideration needed to be given to the design of the teaching intervention. It is essential that a well-planned and appropriately-resourced teaching scheme is produced in addition to the microworld activity itself. If this principle is not followed one runs the risk of committing what Papert (1987) would term *technocentrism*: a preoccupation with the technology itself whilst ignoring the wider practice of the technology being used.

Research suggests (*see* Rieber, 2004) that facilitation of access to programming activities is vital to the success of microworld-based learning approaches. Some responses in WP3 highlighted a need to increase access further with respondent 23 commenting: '[the activities] could be a bit simpler because sometimes they are really confusing' (*see* section 6.3.2.i).

Following this data, new offline teaching aids were produced for WP4 with competition briefs and instruction sheets for WP5 (*see* appendices F and G). Further, WP5 (*see* section 8.1.1) made use of an *incremental* approach (*see* Brusilovsky et. al, 1994) to introducing programming. WP5 uses a series of tutorials that exemplify different aspects of computation rather than the entire range at once. The tutorials were of an increasing level of difficulty and explained the functionality of the microworld in a phased way with partially-populated activities. This was preferable to the *tabula rasa* approach that overwhelmed some learners in the earlier studies.

R3. Ensure that microworlds provide sufficient challenge as part of mixed-attaining teaching

Rieber's (2004) criterion for a microworld design that is accessible to beginners is accompanied with a requirement for it to be intrinsically motivating. This means, in simple terms, that it should employ a 'high ceiling' - as well as a 'low floor'- to enable other learners to develop their ideas with increasing complexity. The findings in WP2 (*see* section 5.3.2.i) derived from low-attaining academic streams reveal an improvement in computational thinking that was less statistically significant than that which emerged in the mixed-attaining settings of WP3 (*see* section 6.3.1.ii) and WP4 (*see* section 7.3.1.i). The results indicate that the microworld intervention had a greater impact on computational thinking development in the mixed-attaining settings.

In WP3, respondent 16 commented that the microworld could be improved 'by teaching us how the […] blocks […] are made' (*see* section 6.3.2.iii). Indeed, in WP5 it was pleasing to see learners like Ethan constructing their own programming blocks (*see* section 8.3.2.iii). The extensibility offered by the Snap!/BYOB programming environment allowed some learners to develop their own custom blocks for generating poems. Other learners, meanwhile, did not progress beyond using the existing custom blocks they were provided with. The combination of access and challenge ('low floor'/'high ceiling') provided by the custom block functionality makes Snap! particularly suited to microworlds-based learning approaches.

R4. Use pair programming as a pedagogical technique for working with microworlds

WP3 led to the adoption of a pair programming format in the latter WPs following the recommendations made by participants in their learning journals. Learners suggested that they needed to 'talk more' about the work that is being carried out (*see* section 6.3.2.iii). In order to minimize the issue of access, a pair programming approach opened up possibilities of peer learning and co-construction of knowledge.

The benefits of group work have been advocated by educational thinkers (e.g. Vygotsky, 1978). There is also evidence from previous research which demonstrates that a pair programming approach can expedite the process of solving problems and lead to the co-construction of technical programming skills (*see* Cockburn and Williams, 2000; Chong et al., 2005). It was decided, therefore, to use this approach from WP3 onwards.

It was pleasing to see evidence of co-construction of knowledge surfacing in the qualitative analysis of WP4 with Scarlet and Morgan's conversation about word classes serving as a case in point (*see* section 7.3.2.iii). Based on the evidence gathered here, pair programming provides a productive way of working with microworlds.

R5. Incorporate a design exhibition format into the microworld-based teaching scheme

Provision for a design exhibition was absent in WP3, and learners commented that they would value a way to share their poems with the rest of the class (*see* section 6.3.2.iii). There was a need, in other words, for learners to share their *outcomes* as well as the *process* of creating the poems.

The value of social interactions as part of the learning process is well-supported by research (*see*, for example, Vygotsky, 1962). The format of a design exhibition for learning about programming has been used successfully with school-age children in the US (*see* Resnick et al. 2005b). The format is characterized by an extended computer programming project that takes places over numerous sessions. The task brief needs to be suitably open-ended to enable creative interpretation and culminates at the end of the project with a 'show and tell' session about what has designed.

Following the finding in WP3, the final WP used the Tanaga Challenge brief to provide a context for learners to program their poems alongside a discussion of the code that was used to produce them at the end. Alongside pair programming, a recommendation for educators is to plan a teaching scheme that culminates with a design exhibition format being used.

R6. Choose a programming platform with a customizable user interface

The ways in which learners will interact with a microworld needs to be given due consideration before it is implemented. Snap!/BYOB is split into four distinct areas: the *scripts palette* for choosing blocks, the *scripting pane* for assembling blocks into a program, the *stage* for running the program, the *sprites pane* for dealing with on-screen objects. Learners were not required to interact with the sprites pane as part of this project. With a large volume of information on the screen at any one time, it is necessary to the customize the user interface to make it suitable for learners to interact with.

Concerns were highlighted in WP3 with the readability of the text contained in the list monitors that held the words on the stage. This was later rectified in a subsequent update to Snap! with a 'zoom blocks' feature that enabled a larger font size (*see* section 6.3.2.iv). In addition, some learners in this earlier WP expressed confusion with the layout of Snap! and were overwhelmed with so much information on screen at one time. It was therefore necessary in the later WPs to provide specific instruction on the use of the stage-resizing button for viewing the poetry after it had been generated. If the functionality to make these UI customizations was not included, it is possible that learners would face barriers of access when exploring with the microworld.

R7. Plan opportunities for dialogic teaching when working with microworlds

Monologic discourse in a classroom is seen where the teacher remains in control of classroom talk and is concerned with the transmission of knowledge. The *dialogic* teacher, on the other hand, encourages learners to ask their own questions and take control of discussing issues, thereby 'challeng[ing] the asymmetrical power relations' of classroom talk that traditionally privileges the teacher' (Lyle, 2008, p. 225).

Analysis of the communicative events (CE) in WP5 provided evidence to suggest that dialogic teaching, when used alongside a microworlds-based learning approach, is able to contribute to the construction of new knowledge. In CE9, for example, a question that Ovyaa independently forms and directs to the teacher results in her successfully debugging a problem in the syntax of her computer program. CE10 is a further example showing a conversation that took places between Rokas and James. Rokas, in discussion with inexperienced newcomer James, attempts to scaffold his peer's learning by providing him with a strategy for learning programming as a beginner.

Both CE suggest that talking about the issues encountered when exploring the microworld is an important part of the process of learning within it. The sources of these interactions are not only between peers: learner-teacher communication is equally as important.

R8. Establish a programme for teacher CPD in microworld-based learning approaches

There are potential hurdles for educators to manage if they wish to incorporate aspects of microworld-based learning into their own pedagogy for literacy and computational thinking. First, designing programming activities takes time: spending several hours writing programming code cannot be justified for one or two classes to the detriment of planning time that is afforded to other groups. A second issue is the level of technical expertise that is necessary.

Block-based programming languages may have the potential to save time, and reduce the level of technical expertise required, when developing microworlds in comparison to text-based approaches. This project has produced both quantitative and qualitative data demonstrating improvements in aspects of computational thinking and literacy when incorporating elements of a microworld-based learning approach into classroom practice. A key priority for future development is to build on these positive findings by establishing a CPD programme for school age educators. Further, there should be a strong link between microworlds CPD for existing teachers and new teachers in enrolled in ITE (*see* Furlong, 2015).

One possible outlet for microworlds CPD is provided by the Computing at School (2017) (CAS) group that is run by the British Computer Society (BCS). Aiming to promote and develop the teaching of computing in schools, *CAS* has established a series of regional hubs that are run by school age teachers with the same audience. The idea is that groups of teachers from different schools are able to attend face-to-face meetings in order to share ideas about how to develop the teaching of computing at their institutions. The teacher-researcher of this project is the co-leader of a regional CAS hub that provides a useful research output for disseminating the findings of this action research project.

R9. Consider the opportunities for microworld-based learning in light of school transformation in Wales

Curriculum reorganisation is high on the political agenda in Wales with significant changes taking place from 2021 (Welsh Government, 2016a). Recommendations from the Donaldson (2015) review will see a stronger focus on 'three Cross-curriculum Responsibilities that should be the responsibility of all teachers: literacy; numeracy; and digital competence' (p. 113). Further, the thirteen subject orders of the National curriculum are to be 'streamlined' and replaced with six broader AoLE.

Whilst substantive school transformation presents challenges to educators, it also provides opportunities. Whereas this project has focused on language learning and aspects of literacy development, the potential of the microworlds-based model of learning extends beyond this domain. The WP1 pilot demonstrates the multiplicity of ways in which the microworlds-based learning approach can meaningfully integrate the data and computational thinking strand of the Digital Competence Framework (*see* Welsh Government, 2016b) into future AoLE. The Rocket Maze, Feudal Society and Pyramids of Numbers activities are examples that demonstrate the potential value of microworlds-based learning within future AoLE of mathematics and numeracy, humanities and science and technology. There may also be potential value of microworld-based learning beyond those curriculum areas identified the WP1 trial to other areas such as physics and environmental science.

As the new curriculum is developed, it is important that the education community reflects on *how* this impacts on curriculum delivery and not just *what* is delivered. With curriculum reform attaching a greater focus to learning across (not *just* within) curricular boundaries (*see* Donaldson, 2015), there is a need to explore potential models of teaching and learning that cut across subject disciplines. As educators, there is a danger that new developments such as the Digital Competence Framework will become 'side-lined' or 'bolted-on' to the new curriculum. The findings suggest a means of ensuring that computational thinking is authentically embedded within the transformed curriculum. In this way, the project demonstrates the potential contribution that microworlds-based pedagogy can make to shaping future models of teaching and learning across the new AoLE.

9.2.4 Other Findings

Gender did not figure as a guiding question of the project. The structure of the SPSS data collected in WP3 and WP4, however, means that it is possible to make a comment on gender difference and differences in performance.

For computational thinking in WP3 and WP4 there is a weak level of confidence that girls improved more than boys (43% and 45%, *see* sections 6.3.1.ii and 7.3.1.i). For literacy, there is greater confidence in a gender performance difference (78% and 80%, *see* ibid.).

This data suggests a closer correlation between gender and improvement in literacy than gender and improvement in computational thinking. It is important to note, however, that no statistically significant differences in gender performance for either computational thinking or literacy were seen in these studies. Further research is needed before any relationship between gender and improvements in literacy and computational thinking can be posited.

9.3 Implications

This thesis ends by asking how its findings about microworlds-based teaching approaches may fit within evolving ideas of teaching and learning.

9.3.1 Changing classroom practice

Back in 1980, Papert (1980a) coined the phrase *microworld* to describe how learning happens with Turtle Graphics. He developed a computer-based environment where learners are able to construct on-screen drawings by assembling lines of programming code. The microworld enabled the acquisition of ideas in computational thinking and mathematics to take place *at the same time as* learning to program.

More recently, block-based programming languages such as Scratch (Resnick et al., 2004a) have increased access to programming activities and its influence has been significant amongst educators. Meanwhile, the simplicity of the original Turtle Graphics microworld continues to shape approaches to learning programming. Code.org (2015), for example, has replaced the lines of code with programming blocks whilst the Turtle has been given a 21st century upgrade: Elsa of Arendelle from Disney's movie *Frozen*. The aim is to assemble blocks and change variables in order for Elsa to create a pattern in the ice.

Whilst Papert's work demonstrates how mathematical ideas may be acquired through the act of learning to program, it is the work of Sharples (1985) and Goldenberg and Feurzeig (1987) that developed microworlds for language learning. Building on these foundations, this project establishes the value and efficacy of extensible (i.e. 'build your own blocks') programming tools for reviving the microworld-based approach. It does so with a particular focus on aspects of computational thinking and literacy development.

The microworlds-based approach is rooted in the 1960s - 1970s before becoming more widely known in the 1980s. It should not be ignored by educators as outmoded. The Innovating

Pedagogy 2016 report (Sharples at al., 2016) provides ten innovations that, though already in use, are likely to figure more prominently in classrooms of the future. The fourth of these, design thinking, connects clearly with the values of the microworlds-based learning approach. The report explains *design thinking* by recalling the work of Rowe (1987):

> Rowe describes *design thinking* [my italics] as a series of 'skirmishes' or 'engaged episodes' with the problem at hand, exploring the relationships between form, structure, and technical issues […] Designers work with the constraints of their materials – concrete and glass for an architect, or colours and computer code for a web designer – employing these constraints as resources to make decisions about what will be elegant and what will work in practice (Sharples et al., 2016, p. 23).

Design thinkers, as the *Innovating Pedagogy* report puts it, 'do more than resolve technical problems. They explore how their designs will respond to human needs and interests' (*ibid.*). And this idea of a pluralization of knowledge is shared by both Papert's formulation of a *microworld* and Rowe's articulation of *design thinking.* As a designer in Turtle Graphics, the learner needs to engage with the *technical* issue of how to write a computer program, the *mathematical* issue of angle of turn as well as the *aesthetic* issue of what their graphic should look like. In the poetry microworld-based activities presented here, learners were required to put their technical programming skills to appropriate use in order to generate lines of poetry. In so doing, learners were also required to reflect upon the form and structure of a poetic form, making creative choices about the word types and sentence structures required in order to communicate successfully.

The benefits of block-based programming tools *to the learner* are documented in this thesis. The tools provide an opportunity to engage with complex programming concepts whilst 'lowering the floor' through removing the barrier of syntax (*see* Resnick et al., 2009; Rieber, 2004). Tools such as Snap! enable the educator to create custom block kits that are tailored to specific aspects of learning within a domain of knowledge. And, in comparison to earlier text-based languages, this can be accomplished with relative speed.

With computational thinking set to become part of a third cross-curricular responsibility for all school age educators in Wales, teachers are busy reflecting on how their pedagogy should adapt. Perhaps, given the recent developments in policy in Wales, the time is now right to revive the microworld tradition in Wales.

9.3.2 Policy developments in Wales

The current curriculum in Wales is comprised of thirteen programmes of study – or 'subject orders' – with statutory cross-curricular skills in literacy and numeracy that are developed across subjects (*see* Welsh Government, 2015c; DfES, 2013). This is due to be replaced by six AoLE (areas of learning and experience) with digital competence designated a third area of cross-curricular responsibility (*see* Donaldson, 2015). Progress has already been made with the Digital Competence Framework (Welsh Government, 2016a), albeit in draft form only at present, with the target of 2021 to commence delivery of the new curriculum (*see* Welsh Government, 2015b).

What do these developments tell us about the future for learners and teachers in Wales? The emphasis given to responsibilities that cut across the curriculum, as well the reconfiguration of the subject orders into six broader AoLE, appears to point to a curriculum that is less segmented. Donaldson's report also includes a list of twelve pedagogical principles that should inform teaching and learning in Wales, with principles eight and nine being particularly relevant to this thesis:

> 8. Good teaching and learning ranges within and *across* [*my italics*] Areas of Learning and Experience
>
> 9. Good teaching and learning regularly reinforces Cross-curriculum Responsibilities, including literacy, numeracy and digital competence, and provides opportunities to practise them (Donaldson, 2015).

The activities trialled here provides a possible way of addressing pedagogical principle nine. The mapping to the specific aspects of the literacy framework (DfES, 2013) undertaken in this thesis clearly shows how the activities provide an opportunity to develop literacy. Further, the mapping to Resnick and Brennan's (2012) framework for computational thinking makes a connection to the data and computational thinking strand of the draft digital competence framework (Welsh Government, 2016a).

Donaldson's eighth pedagogical principle states that learning should take place *across* AoLE as well as *within* them. The emphasis this project gives to embedding cross-curriculum responsibilities *across* different curriculum areas links with the former mode of learning identified by Donaldson. This project has demonstrated how programming activities focusing on literacy can be embedded in subjects such as English and drama. This research provides one potential model for the development of literacy and digital competence within the two future AoLE of *languages, literacy and communication* and *expressive arts*.

Finally, it is important to reference a criticism that Donaldson makes of previous reviews of curricula in Wales. He argues that they have advocated particular models of teaching and learning over others by either reasserting instructionism or by proposing discovery-based, constructivist models (*ibid.*, p. 65). He is clear that a blend of approaches – 'a broad repertoire' – is required for teaching and learning to be most effective.

In the literature review, the theoretical position of Rieber (1992) and his formulation of the microworld-based approach as a *guided* discovery model of learning was considered. For Rieber, the microworld provides a bridge between the poles of instructionism and constructionism. The research presented in this project aligns with Rieber's theoretical position by resisting both a 'purely' constructionist model of exploration whilst also avoiding a set of teacher-led programming exercises. Instead, the research shows how an educator can use existing block-based programming tools and customize these appropriately in order to encourage knowledge acquisition with a specific domain. Perhaps, in this way, the microworld-based approach has the potential to fit within the 'broad repertoire' of pedagogies that Donaldson says is needed for Wales to be successful in the future.

9.4 Future Work

This research has contributed to the field of microworlds research by examining the value of extensible block-based programming technologies for the constructive learning of specific aspects of literacy and computational thinking. All work was carried out within realistic classroom settings and with an authentic curriculum context. Additionally, the research has formulated some key recommendations for educators wishing to incorporate aspects of the microworld-based learning approach into their own pedagogy. There are, however, two main issues raised by this research that could be used to guide future work.

First, whilst learners engaged with a Small Basic *text-based* programming activity in two of WPs, the key focus of the studies was extensible *block-based* programming approaches. The classroom time dedicated to experimenting with text-based programming languages was limited. An interesting area of future research is to explore how educators should manage the progression from block-based programming tools to text-based programming tools of increasing complexity.

Second is the question of how to disseminate the findings of the project and assess the potential impact on classroom practice in other schools. The conclusions here provide evidence to support the potential benefits of microworld-based learning approaches within the school where the teacher-researcher is employed. The localization of the findings was an important characteristic of the action research methodology that was used. Research should consider, however, how teacher training and education could incorporate elements of microworld-based learning into their programmes of study. This is particularly relevant in Wales where there is a move to a more flexible approach to subject boundaries as well as the placement of digital competence at the centre of curriculum reform.

References

ACCAC. (2004) *Review of the School Curriculum and Assessment Arrangements 5-16.* Cardiff: Qualifications, Curriculum and Assessment Authority for Wales.

Alexander, R. (2006) *Towards Dialogic Teaching*. 3rd edn. New York: Dialogos.

Al-Khatib, H. (2009) 'How has Pedagogy Changed in a Digital Age? ICT Supported Learning: Dialogic Forums in Project Work,' *European Journal of Open, Distance and E-Learning*, 2(1) [Online]. Available at http://www.eurodl.org/index.php?article=382 (Accessed: 14 February 2012).

Apple, Inc. (2008) *Steve Jobs on App Store*. Available at: http://www.youtube.com/watch?v=dWynAHdVqnQ (Accessed: 02 May 2012).

Bates, M. and Wilson, K. (1981) *'Illiad': Interactive Language Instruction Assistance for the Deaf, Report No. 4771.* Boston, MA: Bolt, Beranek and Newman, Inc.

Bell, J. (1993) *Doing Your Research Project: A Guide for First-Time Researchers in Education and Social Science.* 2nd edn. Buckingham: Open University Press.

Bell, J. (2005) *Doing Your Research Project: A Guide for First-Time Researchers in Education, Health and Social Science*. 4th edn. Kindle format [e-book reader]. Available at: http://www.amazon.co.uk (Accessed 6 November 2013).

Berg, B. (2001) *Qualitative Research Methods for the Social Sciences*. 4th edn. Massachusetts: Allyn and Bacon.

Black, T. (2002) *Understanding Social Science Research*. 2nd ed. London: Sage.

Boden, M. (1994) *Piaget.* 2nd ed. London: Fontana.

Boyle, T. (2004) 'Designing Multimedia e-Learning for Science Education' in Hollman, R. and Scanlon, E. (eds.) *Mediating Science Learning through Information and Communications Technology.* London and New York: RoutledgeFalmer, pp. 103–119.

Braitenberg, V. (2001) *Vehicles: Experiments in Synthetic Psychology*. Cambridge, MA: MIT Press.

Bradshaw, P. (2012) 'Computing at School: An Emergent Community of Practice for a Re-Emergent Subject', *International Conference on Information Communication Technologies in Education*, Rhodes, Greece, 5-7 July. Mytilene, Greece: University of the Aegean [Online]. Available at: http://www.icicte.org/Proceedings2012/Papers/15-2-Bradshaw.pdf (Accessed: 27 October 2013).

Brennan, K. (2009) 'Mind the Gap: Differences Between the Aspirational and the Actual in an Online Community of Learners', *Proceedings of 8th International Conference on Computer Supported Collaborative Learning*, Rhodes, Greece, 8-13 June. Rhodes, Greece: CSCL [Online]. Available at: http://dl.acm.org/citation.cfm?id=1599529 (Accessed: 27 October 2013).

British Broadcasting Corporation (2014) 'Explorative strategies', *BBC Bitesize*. Available at: http://www.bbc.co.uk/schools/gcsebitesize/drama/exploring/explorative_strategiesrev1.shtml (Accessed: 18 April 2017).

British Educational Research Association (2011) *Ethical Guidelines for Educational Research*. London: BERA.

British Sociological Association (2002) *Statement of Ethical Practice for the British Sociological Association.* Durham: BSA.

Brooks, F. (1995) *The Mythical Man-Month*. San Francisco: Wiley.

Bruner, J. (1982) 'The Language of Education', *Social Research: An International Quarterly,* 49(4), pp. 835-853.

Brusilovsky, P., Kouchnirenko, A., Miller, P. and Tomek, I. (1994) 'Teaching Programming to Novices: A Review of Approaches and Tools, *Proceedings of Ed-Media 94 World Conference on Educational Multimedia and Hypermedia*, Vancouver, 25-30 June. Washington DC: ERIC [Online]. Available at: http://files.eric.ed.gov/fulltext/ED388228.pdf (Accessed 18 February 2016).

Burns, J. (2012) 'School ICT to be Replaced by Computer Science Programme', *BBC News,* 15 January [Online]. Available at: http://www.bbc.co.uk/news/education-16493929. (Accessed: 8 June 2012).

Burke, Q. and Kafai, Y. (2010). 'Programming and Storytelling: Opportunities for Learning About Coding and Composition', *Proceedings of the 9th International Conference on Interaction Design and Children*, Barcelona, Spain, 9-12 June. New York: ACM [Online]. Available at: https://www.seas.upenn.edu/~eas285/Readings/IDC_StorytellingAndProgramming.pdf (Accessed: 18 September 2016).

Cambridge Assessment. (2013) *What is literacy? An Investigation into Definitions of English as a Subject and the Relationship Between English, Literacy and 'Being Literate'.* Available at: http://www.cambridgeassessment.org.uk/images/130433-what-is-literacy-an-investigation-into-definitions-of-english-as-a-subject-and-the-relationship-between-english-literacy-and-being-literate-.pdf (Accessed: 10 August 2016).

Christakis, N. and Fowler, J. (2011) *Connected*. London: HarperPress.

Chong, J., Plummer, R., Leifer, L., Klemmer, S., Eris, O. and Toye, G. (2005) 'Pair Programming: When and Why it Works', *Proceedings of the Seventeenth Annual Workshop of the Psychology of Programming Interest Group* (PPIG 2005), Sussex, Brighton, 29 June – 1 July. Sussex: University of Brighton [Online]. Available at: http://www.ppig.org/papers/17th-chong.pdf (Accessed: 22 September 2015).

Cockburn, A. and Williams, L. (2000) 'The Costs and Benefits of Pair Programming', *Proceedings of the First International Conference on eXtreme Programming and Flexible Processes in Software Engineering (XP2000)*, Cagliari, Sardinia, Italy, 21 – 23 June. Calgary, CA: University of Calgary [Online]. Available at: http://collaboration.csc.ncsu.edu/laurie/ Papers/XPSardinia.PDF (Accessed: 22 September 2015).

Code.org (2015) *Hour of Code*. Available at https://code.org/learn (Accessed: 18 February 2016).

Cooper, S., Dann, W. and Pausch, R. (2003) 'Teaching Objects First in Introductory Computer Science', *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education*, Reno, NV, 19-23 February. New York: ACM [Online]. Available at: http://www.stanford.edu/~coopers/ alice/cooper_dann.pdf (Accessed: 27 October 2013).

Cohen, L. and Manion, L. (1994) *Research Methods in Education*. 4th edn. Oxon and New York: Routledge.

Cohen, L., Manion, L. and Morrison, K. (2007) *Research Methods in Education.* 6th edn. Oxon and New York: Routledge.

Cohen, L., Manion, L. and Morrison, K. (2011) *Research Methods in Education*. 7th edn. Kindle format [e-book reader]. Available at http://www.amazon.co.uk (Accessed 6 November 2013).

Computing at School Working Group (2012a) *Computer Science as a School Subject: Seizing the Opportunity*. Available at: http://www.computingatschool.org.uk/data/uploads/ Case%20for%20Computing.pdf (Accessed: 7 July 2012).

Computing at School Working Group (2012b) *A Curriculum Framework for Computer Science and Information Technology*. Available at: http://www.computingatschool.org.uk/data/uploads/Curriculum%20Framework%20for%20CS%20and%20IT.pdf (Accessed: 7 July 2012).

Computing at School Working Group (2012c) *Computer Science: A Curriculum for Schools*. Available at: http://www.computingatschool.org.uk/data/uploads/ComputingCurric.pdf (Accessed: 7 July 2012).

Computing at School (2016) *Computational Thinking*. Available at: http://barefootcas.org.uk/wp-content/uploads/2016/06/ComputationalThinking-Concept-Barefoot-Computing.pdf (Accessed: 10 April 2017).

Computing at School (2017) *CAS Community*. Available at: https://community.computingatschool.org.uk/hubs (Accessed: 18 June 2017).

Cooper, S., Dann, W. and Pausch, R. (2003) 'Teaching Objects First in Introductory Computer Science', *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education*, Reno, NV, 19-23 February. New York: ACM [Online]. Available at: http://www.stanford.edu/~coopers/ alice/cooper_dann.pdf (Accessed: 27 October 2013).

Cooper, S., Lurie, D. and Moskal, B. (2004) 'Evaluating the Effectiveness of a New Instructional Approach', *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education*, Norfolt, VA, 3-7 March. New York: ACM [Online]. Available at: http://www.stanford.edu/~coopers/alice/p75-moskal.pdf (Accessed: 27 October 2013).

Craft, A (2000) *Continuing Professional Development: A Practical Guide for Teachers and Schools.* 2nd edn. London: RoutledgeFalmer.

Creswell, J. (2014) *Research Design: Qualitative, Quantitative and Mixed Methods Approaches*. 4th edn. London: Sage.

Csikszentmihalyi, M. (2004) *Flow and the Foundations of Positive Psychology: The Collected Works of Mihaly Csikszentmihalyi*. Springer: London and New York.

Curzon, P., Dorling, M., Ng, T., Selby, C. and Woollard, J. (2014). *Developing Computational Thinking in the Classroom: A Framework*. Available at: https://academy.bcs.org/sites/academy.bcs.org/files/DevelopingComputationalThinkingInTheClassroomaFramework.pdf (Accessed: 13 November 2016).

Dalgarno, B. (2001) 'Interpretations of Constructivism and Consequences for Computer Assisted Learning', *British Journal of Educational Technology,* 32(2), pp. 183-194.

Delmont, S. (2002) *Fieldwork in Educational Settings.* London and New York: Routledge.

Denning, P. (2010) 'The Great Principles of Computing', *American Scientist*, 98(5), pp. 369 – 372.

Denscombe, M. (1998) *The Good Research Guide for Small-scale Social Research Projects.* Buckingham: Open University Press.

Denscombe, M. (2002) *Ground Rules for Good Research: A 10 Point Guide for Social Researchers.* Buckingham: Open University Press.

Denzin, N. (1978) *The research Act: A Theoretical Introduction to Sociological Methods.* New York: McGraw-Hill.

Department for Children Lifelong Learning and Skills (DCELLS) (2008a) *Design and Technology in the National Curriculum for Wales*. Cardiff: Welsh Assembly Government.

Department for Children Lifelong Learning and Skills (DCELLS) (2008b) *Information and Communication Technology in the National Curriculum for Wales*. Cardiff: Welsh Assembly Government.

Department for Children Lifelong Learning and Skills (DCELLS) (2008c) *Mathematics in the National Curriculum for Wales*. Cardiff: Welsh Assembly Government.

Department for Children Lifelong Learning and Skills (DCELLS) (2008d) *Music in the National Curriculum for Wales*. Cardiff: Welsh Assembly Government.

Department for Children Lifelong Learning and Skills (DCELLS) (2008e) *History in the National Curriculum for Wales*. Cardiff: Welsh Assembly Government.

Department for Children Lifelong Learning and Skills (DCELLS) (2008f) *Science in the National Curriculum for Wales*. Cardiff: Welsh Assembly Government.

Department for Children Lifelong Learning and Skills (DCELLS) (2008g) *English in the National Curriculum for Wales*. Cardiff: Welsh Assembly Government.

Department for Children Lifelong Learning and Skills (DCELLS) (2008h) *Art and Design in the National Curriculum for Wales*. Cardiff: Welsh Assembly Government.

Department for Children Lifelong Learning and Skills (DCELLS) (2008i) *Skills Framework for 3 to 19 Year-olds in Wales.* Cardiff: Welsh Assembly Government.

Department for Children Lifelong Learning and Skills (DCELLS) (2008j) *Personal and Social Education Framework for 7 to 19-Year-olds in Wales.* Cardiff: Welsh Assembly Government.

Department for Education (DfE) (2013) *The National Curriculum in England: Framework Document for Consultation*. London: Crown.

Department for Education and Skills (DfES) (2011) *Improving Schools.* Cardiff: Welsh Government.

Department for Education and Skills (DfES) (2013) *National Literacy and Numeracy Framework*. Cardiff: Welsh Government.

Dillenbourg, P. (2013) 'Design for Classroom Orchestration', *Computers and Education*, 69(1), pp. 485 – 492.

Dimitrov, D. and Rumrill, P. (2003) 'Pretest-Posttest Designs and Measurement of Change', *WORK: A Journal of Prevention, Assessment and Rehabilitation,* 20(2), pp. 159-165.

Donaldson, G. (2015) *Successful Futures: Independent Review of Curriculum and Assessment Arrangements in Wales*. London: Crown.

Douglas, T. (2011) 'Google's Eric Schmidt criticises Education in the UK', *BBC News*, 26 August [Online]. Available at: http://www.bbc.co.uk/news/uk-14683133. (Accessed: 8 June 2012).

Ellis, R. (2003) *Task-based Language Teaching*. Oxford: Oxford University Press.

Ennis, D. (1992) 'Programming and Problem-solving: What have we Learned?', *Journal of Computer Science Education*, 6(4), pp. 17–20.

Ericsson, K. and Simon, H. (1993) *Protocol Analysis: Verbal Reports as Data*. Cambridge, MA: MIT Press.

Estyn. (2010) *Inspection under section 10 of the Schools Inspection Act 1996: School 2.* Cardiff: Estyn.

Estyn. (2011) *Inspection under section 10 of the Schools Inspection Act 1996: School 1.* Cardiff: Estyn.

Estyn. (2012) *The Skills Framework at Key Stage 3: An Evaluation of the Non-statutory Skills Framework for 3 to 19-year-olds at Key Stage 3.* Cardiff: Estyn.

Estyn. (2016) *Inspection under section 10 of the Schools Inspection Act 1996: School 1.* Cardiff: Estyn.

Fleming, M. (2006) 'The Teaching of Language as School Subject: Theoretical Influences', *Languages of Schooling : Towards a Framework for Europe,* Strasbourg, 16 – 18 October. Strasbourg: Council of Europe [Online]. Available at: www.coe.int/t/dg4/linguistic/Source/ LE_Conf06_Report_May07_EN.doc (Accessed: 10 August 2016).

Frayling, C. (2011) *On Craftmanship: Towards a New Bauhaus.* London: Oberon.

Furlong, J. (2015) *Teaching Tomorrow's Teachers: Options for the Future of Initial Teacher Education in Wales* [Online]. Available at: http://gov.wales/docs/dcells/publications/ 150309-teaching-tomorrows-teachers-final.pdf (Accessed: 21 January 2018).

Future Skills Wales. (2003) *Future Skills Wales 2003 Generic Skills Survey Summary Report*. London: FSW Partnership.

Gargarian, G. (1996) 'The Art of Design' in Resnick, M. and Kafai, Y. (eds.) *Constructionism in Practice*. Mahwah, NJ: Lawrence Erlbaum Associates, pp. 125-159.

Gauntlett, D. (2011) *Making is Connecting: The Social Meaning of Creativity, from DIY and Knitting to Youtube and Web 2.0*. Cambridge and Malden, MA: Polity.

Gee, J. 'Literacy, Discourse and Linguistics: Introduction', *Journal of Education*, 171(1), pp. 5 – 17.

Goldenberg, E. and Feurzeig, W. (1987) *Exploring Language with Logo*. Cambridge, MA: MIT Press.

Google Developers. (2016) *Blockly | Google Developers*. Available at: https://developers.google.com/ blockly/ (Accessed: 18 February 2016).

Harel, D. and Feldman, Y. (2004) *Algorithmics: The Spirit of Computing*. 3rd ed. London: Pearson Education.

Hennessy, S., Rojas-Drummond, S., Higham, R., María Márquez, A., Maine, F., María Ríos, R., García-Carrión, R., Torreblanca, O. and José Barrera, M. (2016a) 'Developing a Coding Scheme for Analysing Classroom Dialogue Across Educational Contexts', *Learning, Culture and Social Interaction*, [Online]. Available at: http://dx.doi.org/10.1016/j.lcsi.2015.12.001 (Accessed: 6 April 2016).

Hennessy, S., Rojas-Drummond, S., Higham, R., María Márquez, A., Maine, F., María Ríos, R., García-Carrión, R., Torreblanca, O. and José Barrera, M. (2016b) *Cam-UNAM SEDA Condensed version.* Available at: https://docs.google.com/document/d/ 1tlSw1pFioN68nssE6hpX8T5hGQHZB212XQltY5VfPE/edit?pref=2&pli=1#heading=h.gjdgxs (Accessed: 7 April 2016).

Hennessy, S., Rojas-Drummond, S., Higham, R., María Márquez, A., Maine, F., María Ríos, R., García-Carrión, R., Torreblanca, O. and José Barrera, M. (2016c) *Cam-UNAM Scheme for Educational Dialogue Analysis (SEDA) – Full version.* Available at:https://docs.google.com/document/d/ 1De48c9GoUZKl0JHjquGR0fNhP9NSjZYeUFeL3lYwhTA/edit?pref=2&pli=1 (Accessed: 7 April 2016).

Hopkins, D. (2002) *A Teacher's Guide to Classroom Research.* Buckingham: Open University Press.

Hancock, B. (1998) *Trent Focus for Research and Development in Primary Health Care: An Introduction to Qualitative Research.* Available at: http://faculty.cbu.ca/pmacintyre/ course_pages/ MBA603/MBA603_files/IntroQualitativeResearch.pdf (Accessed: 27 October 2013).

Hmelo-Silver, C. (2004) 'Problem-based Learning: What and How do Students Learn?', *Educational Psychology Review*, 16(3), pp. 235 – 266.

Hopkins, D., (2002) *A Teacher's Guide to Classroom Research.* Buckingham: Open University Press.

Howe, J., Ross, P., Johnson, K., Plane, F. and Inglis, R. (1982a) 'Teaching Mathematics through Programming in the Classroom', *Computers and Education*, 6(1), pp. 85-91.

Howe, J., DuBoulay, B. and Johnston, K. (1982b) *Re-learning Mathematics through LOGO: DAI Research Paper No. 178*. Edinburgh: University of Edinburgh Department of Artificial Intelligence.

Howe, J. (1984) *Learning Middle School Mathematics through LOGO Programming: An Evaluation Programme: DAI Research Paper No. 203*. Edinburgh: University of Edinburgh Department of Artificial Intelligence.

Howland, K. and Good J. (2015) 'Learning to Communicate Computationally with Flip: A Bi-modal Programming Language for Game Creation', *Computers and Education*, 80(1), pp. 224 – 240.

Hughes, C. (1997) *Qualitative and Quantitative Approaches to Social Research*. Available at: http://www2.warwick.ac.uk/fac/soc/sociology/staff/academicstaff/hughes/researchprocess/quantitative_and_qualitative_approaches.docx (Accessed: 4 August 2015).

Hussein, A. (2009) 'The Use of Triangulation in Social Sciences Research: Can Qualitative and Quantitative Methods be Combined?', *Journal of Comparative Social Work*, 1(1), pp. 1 – 12.

ICT Steering Group (2013) *The ICT Steering Group's Report to the Welsh Government*. Available at: http://learning.wales.gov.uk/docs/learningwales/publications/131003-ict-steering-group-report-en.pdf (Accessed: 7 November 2013)

Janesick, V. (1994) 'The Dance of Qualitative Research Design: Metaphor, Methodolatry, and Meaning' in Denzin, N. and Lincoln, Y. (eds.) *Handbook of Qualitative Research.* Thousand Oaks, CA: Sage, pp. 209–235.

Jenkins, C. (2012) 'Microworlds: Building Powerful Ideas in the Secondary School', International Conference on Information Communication Technologies in Education, Rhodes, Greece, 5-7 July. Mytilene, Greece: University of the Aegean [Online]. Available at: http://www.icicte.org/Proceedings2012/Papers/02-2-Jenkins.pdf (Accessed: 28 October 2013).

Jenkins, C. and Longman, D. (2013) 'Learning (through) Programming: Cross-curricular Applications of a Microworlds Learning Approach', Association of Information Technology in Teacher Education Conference, Bedfordshire, UK, 8-10 July. London: ITTE [Online]. Available at: http://www.craigjenkins.net/ResearchFiles/ITTE.pdf (Accessed 28 October 2013).

Jenkins, C. (2015a) 'Poem Generator: A Comparative Quantitative Evaluation of a Microworld-based Learning Approach', Association of Information Technology in Teacher Education Conference, London, UK, 4 July. London: ITTE [Pecha Kucha]. Available at: http://itte.org.uk/wp/wp-content/uploads/2016/04/Craig-Jenkins-Pecha-Kucha-ITTE-2015.pdf (Accessed 29 January 2017).

Jenkins, C. (2015b) A Work in Progress Paper: A Comparative Evaluation of a Microwords-based Learning Approach for Developing Literacy and Computational Thinking in Cross-curricular Contexts, Proceedings of the 10th Workshop in Primary and Secondary Computing Education, London, UK, 9 - 11 November. New York: ACM. Available at: http://dx.doi.org/10.1145/2818314.2818316 (Accessed: 30 December 2016).

Jenkins, C. (2015c) 'Poem Generator: A Comparative Quantitative Evaluation of a Microworld-based Learning Approach for Teaching English', International Journal of Education and Development using Information and Communication Technology, 11(2), pp. 153 - 167.

Available at: http://ijedict.dec.uwi.edu/include/ getdoc.php?id=6464&article=1972&mode=pdf (Accessed 1 September 2015).

Johnson, B. (1993) *Teacher as Researcher. ERIC Digests.* Available at: http://www.ericdigests.org/1993/ researcher.htm (Accessed: 1 November 2012).

Johnson, R. and Onwuegbuzie, A. (2004) 'Mixed Methods Research: A Research Paradigm Whose Time Has Come', *Educational Researcher*, 33(7), pp. 14-26.

Jonassen, D. (1999) 'Constructivist learning environments on the web: engaging students in meaningful learning', Proceedings of the Educational Technology Conference, Singapore, 9 – 11 February. Singapore: EdTech 99. Available at: http://citeseerx.ist.psu.edu/viewdoc/ download?doi=10.1.1.137.618&rep=rep1&type=pdf (Accessed 26 November 2017).

Kafai, Y. (1996a) Learning Design by Making Games: Children's Development of Design Strategies in the Creation of a Complex Computation Artifact in Resnick, M. and Kafai, Y. (eds.) *Constructionism in Practice*. Mahwah, NJ: Lawrence Erlbaum Associates, pp. 71-96.

Kafai, Y. (1996b) 'Electronic Play Worlds: Gender Differences in Children's Construction of Video Games' in Resnick, M. and Kafai, Y. (eds.) *Constructionism in Practice*. Mahwah, NJ: Lawrence Erlbaum Associates, pp. 97–123.

Kafai, Y. and Peppler, K. (2005) *Creative Coding: Programming for Personal Expression.* Available at: http://download.scratch.mit.edu/CreativeCoding.pdf (Accessed: 27 October 2013)

Kafai, Y. and Peppler, K. (2007a) 'From SuperGoo to Scratch: Exploring Creative Digital Media Production in Informal Learning', *Learning, Media and Technology,* 32(6), pp. 149–166.

Kafai, Y. and Peppler, K. (2007b) 'What Videogame Making Can Teach Us About Literacy and Learning: Alternative Pathways into Participatory Culture', *Proceedings of Digital Games Research Association Conference,* Tokyo, Japan, 24–28 September. Tokyo, Japan: DiGRA [Online]. Available at: http://download.scratch.mit.edu/DiGRA07_games_kafai.pdf (Accessed: 27 October 2013).

Kafai, Y., Peppler, K. and Chiu, G. (2007c) 'High Tech Programmers in Low-Income Communities: Creating a Computer Culture in a Community Technology Center', *Proceedings of the Third Communities and Technologies Conference*, East Lansing, MI, 28–30 June. London: Springer [Online]. Available at: http://download.scratch.mit.edu/CandT_scratch_Fin.pdf (Accessed: 27 October 2013).

Kafai, Y. and Peppler, K. (2007d) 'Collaboration, Computation and Creativity: Media Arts Practices in Urban Youth Culture', *Proceedings of Computer Supported Collaborative Learning Conference*, New Brunswick, NJ, 16–21 July. New Brunswick, NJ: Rutgers University [Online]. Available at: http://download.scratch.mit.edu/CSCL07_peppler.pdf (Accessed: 27 October 2013).

Kafai, Y., Desai, S., Peppler, K., Chiu, G. and Moya, J. (2008) 'Mentoring Partnerships in a Community Technology Centre: A Constructionist Approach for Fostering Equitable Service Learning', *Mentoring and Tutoring: Partnerships in Learning*, 16(2), pp. 191–205.

Kemmis, S. and Mctaggart, R. (1981) *The Action Research Planner*. Victoria, Australia: Deakin University Press.

Kemmis, S. and Mctaggart, R. (2000) 'Participatory Action Research' in Denzin, N. and Lincoln, Y. (eds.) *Handbook of Qualitative Research*. 2nd edn. Thousand Oaks, CA: Sage, pp. 567-607.

Khan Academy (2010) Available at: http://www.khanacademy.org/ (Accessed: 29 July 2012].

Lacey, C. (1976) ‘Problems of Sociological Fieldwork: a Review of the Methodology of “Hightown Grammar”’ in Shipman, M. (ed.) *The Organisation and Impact of Social Research.* London: Routledge and Kegan Paul, pp. 63–88.

Lanier, J. (2011) *You Are Not a Gadget*. London and New York: Penguin.

Lanier, J. (2013) *Who Owns the Future?* London and New York: Penguin.

Lankshear, C. and Knobel, M. (2013) ‘Introduction: Social and Cultural Studies of New Literacies from an Educational Perspective’ in Lankshear, C. and Knobel, M. (eds.) *A New Literacies Reader: Educational Perspectives*. New York: Peter Lang, pp. 1 – 19.

Lawton, D. and Gordon, V. (1996) *Dictionary of Education*. London: Hodder and Staughton.

Laurillard, D. (2009) ‘The Pedagogical Challenges to Collaborative Technologies’, *International Journal of Computer-Supported Collaborative Learning*, 4(1), pp. 5–20.

Laurillard, D. (2002) *Rethinking University Teaching: A Conversational Framework for the Effective Use of Learning Technologies*. 2nd edn. RoutledgeFalmer: London and New York.

Laurillard, D. (2012) *Teaching as a Design Science: Building Pedagogical Patterns for Learning and Technology.* Routledge: Oxon and New York.

LEGO Education. (2011) *LEGO Education WeDo Construction Set* [Online]. Available at: http://cache.lego.com/r/sc/-/media/lego%20education/home/images/products/wedo/ts.20100922t104512.9580_713x380_mainproduct.png (Accessed: 5 June 2012).

Lessig, L. (2004) *Free Culture: The Nature and Future of Creativity.* London and New York: Penguin.

Lewin, K. (1946) ‘Action Research and Minority Problems’, *Journal of Social Issues*, 2(4), pp. 34-36.

Lifelong Kindergarten Group (2012a) *Languages: Scratch Documentation Site*. Available at: http://info.scratch.mit.edu/Languages/ (Accessed: 4 June 2012).

Lifelong Kindergarten Group. (2012b) *WeDo and Scratch Reference: Scratch Documentation Site.* Available at: http://info.scratch.mit.edu/WeDo/Reference/ (Accessed: 5 June 2012).

Longman, D. (2011) *LOGO Microworlds*. Available at: http://davidlongman.com/logo/logo_microworlds.htm (Accessed: 5 June 2012).

Lyle, S. (2008) ‘Dialogic Teaching: Discussing Theoretical Contexts and Reviewing Evidence from Classroom Practice’, *Language and Education*, 22(3), pp. 222-240.

Maeda, J. (2004) *Creative Code*. New York: Thames and Hudson Inc.

Macefield, R. (2007) ‘Usability Studies and the Hawthorne Effect’, *Journal of Usability Studies*, 2(3), pp. 145-154.

Mayer-Schönberger, V. (2009) *Delete: The Virtue of Forgetting in a Digital Age*. Oxfordshire and Princeton, NJ: Princeton University Press.

McGowan, H. (2011) ‘Planning a Comparative Experiment in Educational Settings’, *Journal of Statistics Education*, 19(2), pp. 1-19.

Microsoft Corporation (2015) *About Microsoft Small Basic*. Available at: http://smallbasic.com/about.aspx (Accessed 28 July 2015).

MIT Media Lab (2012a) *Clubhouse History: Intel Computer Clubhouse Network*. Available at: http://www.computerclubhouse.org/content/clubhouse-history (Accessed 4 June 2012).

MIT Media Lab (2012b) *Custom Block Types | ScratchEd*. Available at: http://scratched.gse.harvard.edu/discussions/scratch-20/custom-block-types (Accessed 7 May 2016).

MIT Media Lab (2016) *Scratch 2.0 – Scratch Wiki*. Available at: https://wiki.scratch.mit.edu/wiki/Scratch_2.0 (Accessed 7 May 2016).

Moodle Trust (2013a) *Moodle.org: Open-Source Community-Based Tools for Learning*. Available at: http://www.moodle.org/ (Accessed 4 January 2014).

Moodle Trust (2013b) *Activities – MoodleDocs*. Available at: http://docs.moodle.org/26/en/Activities (Accessed 4 January 2014).

Mönig, J. and Harvey, B. (2009) 'Bringing "No Ceiling" to Scratch: Can One Language Serve Kids and Computer Scientists?', *Constructionism 2010*, Paris, France, 16-21 August. Paris, France: AUP [Online]. Available at: http://www.eecs.berkeley.edu/~bh/BYOB.pdf (Accessed 29 October 2013).

Mönig, J. and Harvey, B. (2011) *Build Your Own Blocks.* Available at: http://byob.berkeley.edu/ (Accessed: 27 December 2011).

Monroy-Hernández, A. (2007) 'ScratchR: Sharing User-Generated Programmable Media', *Proceedings of 6th International Conference on Interaction Design and Children,* Aalborg, Denmark, 6-8 June. Aalborg, Denmark: IDC07 [Online]. Available at: http://info.scratch.mit.edu/sites/infoscratch.media.mit.edu/files/file/idc07.pdf (Accessed: 27 October 2013).

Monroy-Hernández, A. (2009) 'Designing a Website for Creative Learning', *Proceedings of Web Science Conference*, Athens, Greece, 18–20 March. Athens, Greece: WebSci'09 [Online]. Available at: http://journal.webscience.org/253/ (Accessed: 27 October 2013).

Monroy-Hernández, A., Olson, K., and Hill, B. (2010a) 'Responses to Remixing on a Social Media Sharing Website', *Proceedings of Fourth International AAAI Conference on Weblogs and Social Media*, Washington, D. C, 23–26 May. Washington, D. C.: ICWSM-10 [Online]. Available at: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM10/paper/view/1533 (Accessed: 27 October 2013).

Monroy-Hernández, A. (2010b) 'Cooperation and Attribution in an Online Community of Young Creators', *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work*, Savannah, Georgia, 6–10 February. New York: ACM [Online]. Available at: http://research.microsoft.com/enus/um/redmond/groups/connect/cscw_10/docs/p469.pdf (Accessed: 27 October 2013).

Monroy-Hernández, A., Hill, B., Gonzalez-Rivero, J. and Boyd, D. (2011) 'Computers Can't Give Credit: How Automatic Attribution Falls Short in an Online Remixing Community', *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems,* Vancouver, Canada, 7–12 May. Vancouver, Canada: CHI'11 [Online]. Available at: http://info.scratch.mit.edu/sites/ infoscratch.media.mit.edu/files/file/monroy-hernandez_et_al_chi2011.pdf (Accessed: 27 October 2013)

Morozov, E. (2013) *To Save Everything, Click Here: Technology, Solutionism and the Urge to Fix Problems That Don't Exist.* London and New York: Penguin.

Muijs, D. (2011) *Doing Quantitative Research in Education with SPSS,* 2nd ed. London: Sage.

Naughton, J. (2012) *From Gutenberg to Zuckerberg: What you Really Need to Know About the Internet*. London: Quercus.

O'Donoghue, T. and Punch, K. (2003) *Qualitative Educational Research in Action: Doing and Reflecting.* London and New York: Routledge.

Open University Course E811 (1988) *Educational Evaluation.* Milton Keynes: Open University Educational Enterprises.

Organisation for Economic Co-operation and Development (2008) *21st Century Learning: Research, Innovation and Policy.* Available at: http://www.oecd.org/site/educeri21st/40554299.pdf (Accessed: 17 December 2017).

The Oxford English Minidictionary, 2nd ed. (1995) Oxford: Oxford University Press.

Papert, S. (1972) 'A Computer Laboratory for Elementary Schools', *Computers and Automation*, 21(6), pp. 19-23.

Papert, S. (ca. 1980) *Constructionism vs. Instructionism.* Available at: http://www.papert.org/articles/ const_inst/const_inst1.html (Accessed: 25 December 2011).

Papert, S. (1980a) *Mindstorms: Children, Computers, and Powerful Ideas*. New York: BasicBooks.

Papert, S. (1980b) 'Teaching Children Thinking' in Taylor, R. (ed.) *The Computer in School: Tutor, Tool, Tutee*. New York: Teachers College Press, pp. 161-176.

Papert, S. (1980c) 'Teaching Children to be Mathematicians' in Taylor, R. (ed.) *The Computer in School: Tutor, Tool, Tutee*. New York: Teachers College Press, pp. 177-196.

Papert, S. (1980d) 'Personal Computing and its Impact on Education' in Taylor, R. (ed.) *The Computer in School: Tutor, Tool, Tutee*. New York: Teachers College Press, pp. 197-202.

Papert, S. (1980e) 'Computer-based Microworlds as Incubators for Powerful Ideas' in Taylor, R. (ed.) *The Computer in School: Tutor, Tool, Tutee*. New York: Teachers College Press, pp. 203-210.

Papert, S. (1987) 'Information Technology and Education: Computer Criticism Vs. Technocentric Thinking', *Educational Researcher*, 16(1), pp. 22–30.

Papert, S. (1993a) *The Children's Machine: Rethinking School in the Age of the Computer*. New York: BasicBooks.

Papert, S. (1993b) *Mindstorms: Children, Computers, and Powerful Ideas*. 2nd ed. New York: BasicBooks.

Papert, S. (1996) 'An Exploration in the Space of Mathematics Educations', *International Journal of Computers for Mathematical Learning*, 1(1), pp. 95–123.

Papert, S. (1999) 'Papert on Piaget', *Time*, 29 March, p. 105.

Papert, S. (2008a) *Constructionism: A New Opportunity for Elementary Science Education.* NSF Research Grant. Available at https://nsf.gov/awardsearch/showAward?AWD_ID=8751190 (Accessed: 10 December 2017).

Papert, S. (2008b) *Hard Fun*. Available at http://www.papert.org/articles/HardFun.html. (Accessed: 5 June 2012).

Papert, S. and Harel, I. (1991) 'Situating Constructionism' in Papert, S. and Harel, I. (eds.) *Constructionism*. Norwood, NJ: Ablex, pp. 1–11.

Pea, R. (1984) 'Symbol Systems and Thinking Skills', *Proceedings of 1984 National Logo Conference, Cambridge, MA,* 26–29 June. Cambridge, MA: MIT [Online]. Available at: http://www.stanford.edu/ ~roypea/RoyPDF%20folder/A19_Pea_84.pdf (Accessed: 27 October 2013).

Pea, R. and Kurland, D. (1985) 'Children's Mental Models of Recursive Logo Programs', *Journal of Educational Computing Research,* 1(2), pp. 235–243.

Pea, R., Kurland, D., Clement, C. and Mawby, R. (1987) 'Mapping the Cognitive Demands of Learning to Program' in Pea, R. and Sheingold, K. (eds.) *Mirrors of Minds: Patterns of Experience in Educational Computing.* Norwood, NJ: Ablex, pp. 103–127.

Pea, R. (1987) 'Logo Programming and Problem Solving' in O'Shea, T. and Scanlon, E. (eds.) *Educational Computing: An Open University Reader*. London: John Wiley and Sons, pp. 155–160.

Pearson Education (2009) *SuccessMaker – Product Overview.* Available at: http://www.pearsonschool.com/index.cfm?locator=PSZkAe (Accessed: 29 July 2012).

The Playful Invention Company (2010a) *Getting Started with Picoboards*. Available at: http://www.picocricket.com/pdfs/Getting_Started_With_PicoBoards.pdf (Accessed: 4 June 2012).

The Playful Invention Company (2010b) *PicoBoard – Sensor Board That Works with MIT's Scratch* [Online]. Available at: http://www.picocricket.com/picoboard_images/pbarrowsusb.jpg (Accessed: 5 June 2012).

The Playful Invention Company (2010c) *Meet the PicoCricket*. Available at: http://www.picocricket.com/ pdfs/Meet_PicoCricket.pdf (Accessed: 6 June 2012).

Plonka, L., Segal, J., Sharp, H., Van der Linden, J. (2011) 'Collaboration in Pair Programming: Driving and Switching', *Proceedings of 12th International Conference on Agile Software Development*, Madrid, Spain, 10-13 May. Berlin: Springer [Online]. Available at: https://doi.org/10.1007/978-3-642-20677-1_4 (Accessed: 2 December 2017).

Pound, L. (2006) *How Children Learn: From Montessori to Vygotsky – Educational Theories and Approaches Made Easy*. London: Practical Pre-School Books.

Punch, K. (2005) *Introduction to Social Research: Quantitative and Qualitative Approaches.* 2nd edn. London: Sage.

Ramsay, S. (2011) *Reading Machines: Towards an Algorithmic Criticism.* Champaign, IL: University of Illinois Press.

Resnick, M. (1991) 'Xylophones, Hamsters, and Fireworks: The Role of Diversity in Constructionist Activities' in Papert, S. and Harel, I. (eds.) *Constructionism*. Norwood, NJ: Ablex, pp. 151–158.

Resnick, M. and Kafai, Y. (1996) 'Introduction' in Resnick, M. and Kafai, Y. (eds.) *Constructionism in Practice*. Mahwah, NJ: Lawrence Erlbaum Associates, pp. 1-8.

Resnick, M. (1997) *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel* Microworlds. Massachusetts: MIT.

Resnick, B., Petrich, B., Wilkinson, K., Rusk, N., Smith, B., Willow, D. and Mikhak, B. (2000a) *The PIE Network: Promoting Science Inquiry and Engineering through Playful Invention and Exploration with New Digital Technologies.* Available at: http://llk.media.mit.edu/ papers/pie/. (Accessed: 6 June 2012).

Resnick, M., Berg R. and Eisenberg, M. (2000b) 'Beyond Black Boxes: Bringing Transparency and Aesthetics Back to Scientific Investigation', *Journal of the Learning Sciences*, 9(1), pp. 7–30.

Resnick, M. (2002) 'Rethinking Learning in the Digital Age' in Kirkman, G. (ed.) *The Global Information Technology Report: Readiness for the Networked World*. Oxford: Oxford University Press, pp. 32–37.

Resnick, M., Kafai, Y., Maloney, J., Burd, L., Rusk, N. and Silverman, B. (2004a) 'Scratch: A Sneak Preview', *Second International Conference on Creating, Connecting and Collaborating through Computing,* Kyoto, Japan, 29–30 January. Kyoto, Japan: C5'04 [Online]. Available at: http://llk.media.mit.edu/projects/scratch/ScratchSneakPreview.pdf (Accessed: 27 October 2013).

Resnick, M. (2004b) *Edutainment? No Thanks. I Prefer Playful Learning*. Available at: http://web.media.mit.edu/~mres/papers/edutainment.pdf (Accessed: 5 June 2012).

Resnick, M. Rusk, N. and Berg, R. (2005a) *Rethinking Robotics: Engaging Girls in Creative Engineering.* Available at: http://web.media.mit.edu/~mres/papers/Rethinking-Robotics-final.pdf (Accessed: 6 June 2012).

Resnick, M. Rusk, N., Berg, R. and Pezalla-Granlund, M. (2005b) 'Rethinking Robotics: Learning through Creative Engineering', *Proceedings of 2005 Association of Science – Technology Centres Conference*, Richmond, VA, 15–18 October. Washington, D. C.: ASTC [Online]. Available at: http://llk.media.mit.edu/projects/pie/Rethinking-Robotics-Ideas.pdf (Accessed 27 October 2013).

Resnick, M., Rusk, N., Berg, R. and Pezalla-Granlund, M. (2008a) 'New Pathways into Robotics: Strategies for Broadening Participation', *Journal of Science Education and Technology,* 17(1), pp. 59 – 69.

Resnick, M., Kafai, Y., Maloney, J., Peppler, K. and Rusk, N. (2008b) 'Programming by Choice: Urban Youth Learning Programming with Scratch', *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education*, Portland, Oregon, 12–15 March. New York: ACM [Online]. Available at: http://web.media.mit.edu/~mres/papers/sigcse-08.pdf (Accessed: 27 October 2013).

Resnick, M. and Monroy-Hernández, A. (2008c) 'Empowering Kids to Create and Share Programmable Media', *Interactions*, 15(2), pp. 50–53.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E. Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B. and Kafai, Y. (2009) 'Scratch: Programming for All', *Communications of the ACM*, 52(11), pp. 60-67.

Resnick, M. and Brennan, K. (2012) 'New Frameworks for Studying and Assessing the Development of Computational Thinking', *Proceedings of the 2012 Annual Meeting of the American Educational Research Association*, Vancouver, Canada, 13-17 April. Washington, DC: AERA [Online]. Available at: http://web.media.mit.edu/~kbrennan/files/Brennan_Resnick_AERA2012_CT.pdf (Accessed 28 October 2013).

Resnick, M., Bers, M., Silverman, B., Bontá, P., Kazakoff, E. and Flannery, L. (2013) 'Designing ScratchJr: Support for Early Childhood Learning Through Computer Programming', *Proceedings of Interaction Design and Children*, New York, NY, 24–27 June. New York, NY: ACM Press [Online]. Available at: http://ase.tufts.edu/DevTech/publications/scratchjr_idc_2013.pdf (Accessed: 27 October 2013).

Resnick, M. (2016) *Designing for Wide Walls*. Available at: https://design.blog/2016/08/25/mitchel-resnick-designing-for-wide-walls/. (Accessed: 17 January 2018).

Richards, J. and Reppen R. (2015) 'Towards a Pedagogy of Grammar Instruction', *RELC Journal of Language Teaching and Research*, 45(1), pp. 5 – 25.

Rieber, L. (1992) 'Computer-Based Microworlds: A Bridge Between Constructivism and Direct Instruction', *Educational Technology Research and Development,* 40(1), pp. 93–106.

Rieber, L. (1996) 'Seriously Considering Play: Designing Interactive Learning Environments Based on the Blending of Microworlds, Simulations, and Games', *Educational Technology Research and Development,* 44(2), pp. 43–58.

Rieber, L. (2004) 'Microworlds' in Donassen, D. (ed.) *Handbook of Research for Educational Communications and Technology.* 2nd ed. Mahwah, NJ: Lawrence Erlbaum Associates, pp. 583-603.

Robinson, K. (2009) 'How Schools Stifle Creativity', TedTalk Tuesdays on *CNN*, 3 November [Online]. Available at: http://edition.cnn.com/2009/OPINION/11/03/robinson.schools.stifle.creativity/index.html (Accessed: 26 December 2017).

Robson, S. (2006) *Developing Thinking and Understanding in Young Children: An Introduction for Students*. Oxon and New York: Routledge.

Rowe, P. (1987) *Design Thinking.* Cambridge, MA: MIT Press

The Royal Society (2012a) *Shut Down or Restart? The Way Forward for Computing in Schools.* London: The Royal Society.

The Royal Society (2012b) *Shut Down or Restart? The Way Forward for Computing in Schools: Executive Summary.* London: The Royal Society.

Rushkoff, D. (2010) *Program or be Programmed: Ten Commands for a Digital Age*. New York: OR Books.

Selby C. and Woollard, J. (2013) *Computational Thinking: The Developing Definition*. Available at http://eprints.soton.ac.uk/356481/ (Accessed: 13 November 2016).

Selwyn, N. (2011) *Education and Technology: Key Issues and Debates.* London and New York: Continuum.

Sennett, R. (2009) *The Craftsman.* London and New York: Penguin.

Sharples, M. (1985) *Cognition, Computers and Creative Writing*. West Sussex: Ellis Horwood.

Sharples, M. (1997) 'Storytelling by computer', *Digital Creativity* 8(1), pp. 20 – 29.

Sharples, M. (1999) *How we Write: Writing as Creative Design*. London and New York: Routledge.

Sharples, M., de Roock , R., Ferguson, R., Gaved, M., Herodotou, C., Koh, E., Kukulska-Hulme, A., Looi, C-K, McAndrew, P., Rienties, B., Weller, M., Wong, L. H. (2016) *Innovating Pedagogy 2016: Open University Innovation Report 5*. Milton Keynes: The Open University.

Simmons, M. and Cope, P. (1993) 'Angle and Rotation: Effects of Different Types of Feedback on the Quality of Response', *Educational Studies in Mathematics*, 24(2), pp. 163–176.

Strauss, A. (1989) *Qualitative Analysis for Social Scientists*. Melbourne and New York: Cambridge University Press.

Strauss, A. and Corbin, J. (1998) *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory.* 2nd ed. London and California: Sage.

Terrell, S. (2012) 'Mixed-Methods Research Methodologies', *The Qualitative Report*, 17(1), pp. 254-280.

Tanner, H., Jones, S. and Lewis, H. (2011) 'Metacognition in the Foundation Phase: Using VSRD to Help Young Children Talk about their Thinking', *The Welsh Journal of Education*, 15(1), pp. 66–79.

Taylor, R. (1980) 'Introduction' in Taylor, R. (ed.) *The Computer in School: Tutor, Tool, Tutee*. New York: Teachers College Press, pp. 1–10.

Taylor, R. (2003) 'Reflections on the Computer in the School', *Contemporary Issues in Technology and Teacher Education*, 3(2), pp. 253 – 274.

Trickett, S. and Trafton, J. (2009) 'A Primer on Verbal Protocol Analysis' in Schmorrow, D., Cohn, J. and Nicholson, D. (eds.) *The PSI Handbook of Virtual Environments for Training and Education*. Vol 1. Westport, CT: Praeger Security International, pp. 332- 346.

Turkle, S. and Papert, S. (1991) 'Epistemological Pluralism: Styles and Voices within the Computer Culture' in Papert, S. and Harel, I. (eds.) *Constructionism*. Norwood, NJ: Ablex, pp. 161-193.

Turkle, S. (2007) 'What Makes an Object Evocative?' in Turkle, S. (ed.) *Evocative Objects: Things We Think With*. Cambridge, MA: MIT Press, pp. 307–326.

Vasagar, J. (2012) ‘Michael Gove to Scrap “Boring” IT Lessons’ *The Guardian*, 11 January [Online]. Available at: http://www.guardian.co.uk/politics/2012/jan/11/ michael-gove-boring-it-lessons (Accessed: 20 August 2012).

Vee, A. (2013) ‘Understanding Computer Programming as a Literacy’, *Literacy in Composition Studies*, 1(2), pp. 42 – 64.

Vickers, P. (2008) *How to Think like a Programmer*. London: Cengage.

Vygotsky, L. (1962) *Thought and Language*. Cambridge, MA: MIT Press.

Vygotsky, L. (1978) *Mind and Society: The Development of Higher Mental Processes*. Cambridge, MA: Harvard University Press.

Wang, F. and Hannafin, M. (2005) ‘Design-based Research and Technology-enhanced Learning Environments’, *Education Technology Research and Development*, 53(4), pp. 5 – 23.

Weathington, B., Cunningham, C. and Pittenger, D. (2010) *Research Methods for the Behavioral and Social Sciences*. Hoboken, NJ: John Wiley and Sons.

Webb, R. (2016) ‘Knowledge about grammar: Implications for pre-service TESOL education’, University of South Wales Annual Postgraduate Student Presentation Day, Trefforest, Pontypridd, 1 June. Pontypridd: University of South Wales.

Wegerif, R. and Mercer, N. (1997) ‘A Dialogical Framework for Analysing Peer Talk’ in Wegerif, R. and Scrimshaw, P. (eds.) *Computers and Talk in the Primary Classroom.* Clevedon: The Language and Education Library: 12, pp. 49 – 61.

Weintrop, D., Holbert, N., Horn, M. and Wilensky, U. (2012) ‘Redefining Constructionist Video Games: Marrying Constructionism and Video Game Design’, *Proceedings of Constructionism: Theory, Practice and Impact,* Athens, Greece, 21–25 August. Athens: University of Athens [Online] Available at: http://ccl.northwestern.edu/papers/2012/ 645-649_BP_68_Weintrop.pdf (Accessed: 27 October 2013).

Weintrop, D. and Wilensky, U. (2013) ‘RoboBuilder: A Computational Thinking Game’, *Proceedings of the 44th SIGCSE Technical Symposium on Computer Science Education*, Denver, CO, 6–9 March. New York: ACM [Online]. Available at: http://ccl.northwestern.edu/papers/2013/ RB_Comp_Thinking_Game.pdf (Accessed: 27 October 2013)

Welsh Government (2011) *Raising School Standards* [Online]. Available at: http://gov.wales/docs/dcells/ publications/110629raisingschoolstandardsen.pdf (Accessed: 8 August 2016).

Welsh Government (2015a) *Successful Futures: Independent Review of Curriculum and Assessment Arrangements in Wales* [Online]. Available at: http://gov.wales/newsroom/educationandskills/ 2015/10323651/?lang=en (Accessed: 28 July 2015).

Welsh Government (2015b) *A Curriculum for Wales – A Curriculum for Life* [Online]. Available at: http://gov.wales/docs/dcells/publications/ 151021-a-curriculum-for-wales-a-curriculum-for-life-en.pdf (Accessed: 28 November 2016).

Welsh Government (2015c) *Learning Wales: Key Stage 2 – 4* [Online]. http://learning.gov.wales/ resources/collections/key-stages-2-4?lang=en (Accessed: 28 November 2016).

Welsh Government (2016a) *Learning Wales: Digital Competence Framework* [Online]. Available at: http://learning.gov.wales/resources/browse-all/dcf-questionnaire/?lang=en (Accessed: 23 July 2016).

Welsh Government (2016b) *Draft Digital Competence Framework* [Online]. http://learning.gov.wales/docs/learningwales/publications/ 160611-draft-digital-competence-framework.xlsx (Accessed: 8 August 2016).

Whitebread, D., Jameson, H. and Basilio, M. (2015) 'Play Beyond the Foundation Stage: Play, Self-Regulation and Narrative Skills' in Moyles, J. (ed.) *The Excellence of Play*. 4th edn. Maidenhead: Open University Press, pp. 84 – 93.

Whitebread, D. and Basilio, M. (2015) *Playful Learning – Building Stories Together to Inspire Young Writers: A Teacher Handbook*. Available at: https://www.educ.cam.ac.uk/centres/pedal/plans/ PLaNS%20Handbook_Print-1.pdf (Accessed 12 March 2017).

Willis, I. (2013) *Originality in Doctoral Research: What is it Exactly?* Available at: http://educationaldevelopment.liverpool.ac.uk/2013/03/26/originality-in-doctoral-research-what-is-it-exactly/ (Accessed: 10 November 2013).

Wing, J. (2006) 'Computational Thinking', *Communications of the ACM*, 49(3), pp. 33–35.

Wood, D., Bruner, J. and Ross, G. (1976) 'The role of tutoring in problem solving', *Journal of Child Psychiatry and Psychology*, 17(2), pp. 89 – 100.

Zagal, J. and Bruckman, A. (2005) 'From Samba Schools to Computer Clubhouses: Cultural Institutions as Learning Environments', *Convergence*, 11(1), pp. 98 – 105.

Zimmerman, D. and Wieder, D. (1977) 'The Diary-Interview Method', *Urban Life*, 5(4), pp. 479–499.

Appendices

All appendices appear within the specified directories on the attached data disc and are accompanied by a brief description.

When the data disc is opened, each digital appendix appears as a folder (*see* figure 23).

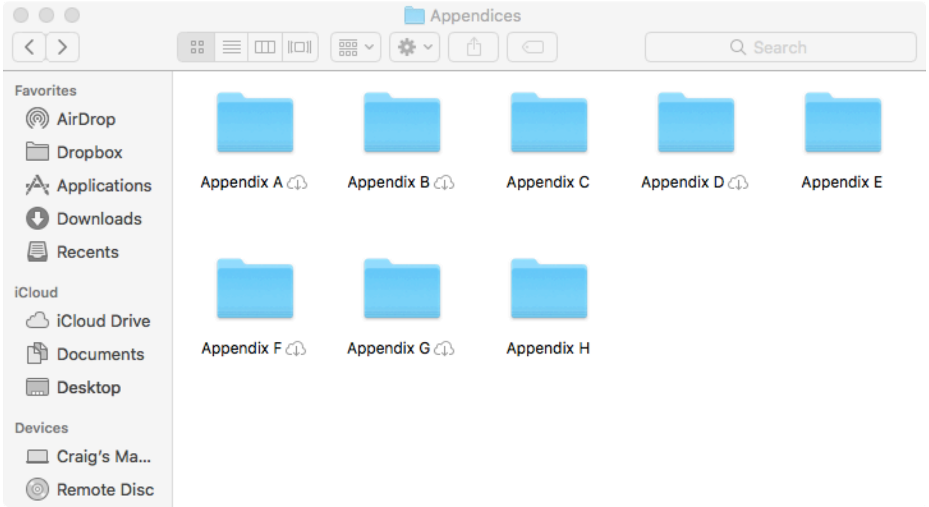


Figure 23: Navigating the data disc

Each folder contains the digital files that are listed in the directory structure for each appendix (*see* figure 24).

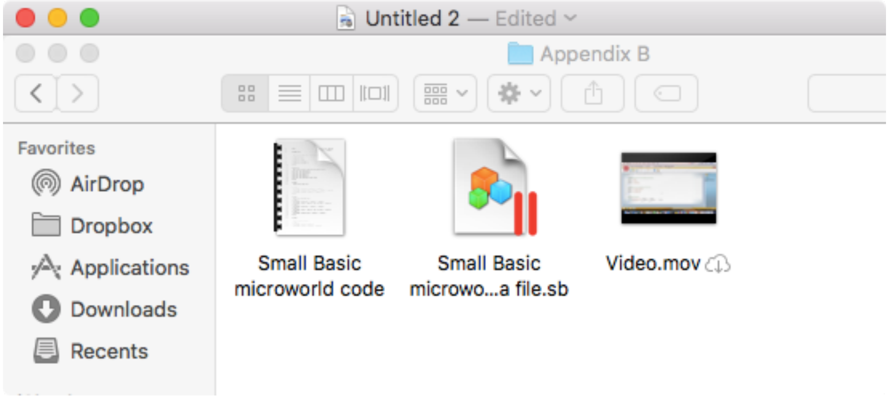


Figure 24: Data disc directory

Where appropriate, text-based appendices appear here in print form. In the interest of readability, text-based appendices covering several pages appear in an abridged form, with unabridged versions appearing on the data disc.

Appendix A: Microworlds Activity Pack and BYOB Data Files for WP1

Description

The Microworlds Activity Pack PDF file provides subject-specific background knowledge, task instructions, computational thinking and curricular links for each of the activities trialled in the pilot study. The YPR data files for viewing the microworld activities can be opened by downloading *BYOB* from *http://snap.berkeley.edu/old-byob.html.* Video demonstrations of each microworld activity are also provided.

Directory/Filename

/ Appendix A / Microworlds Activity Pack.PDF
/ Appendix A / BYOB and Video Files / Feudal Society / Feudal Society.YPR
/ Appendix A / BYOB and Video Files / Feudal Society / Video.MOV
/ Appendix A / BYOB and Video Files / Pyramids of Numbers / Pyramids of Numbers.YPR
/ Appendix A / BYOB and Video Files / Pyramids of Numbers / Video.MOV
/ Appendix A / BYOB and Video Files / Rocket Maze / Rocket Maze 1 - Sequences.YPR
/ Appendix A / BYOB and Video Files / Rocket Maze / Rocket Maze 2 - Inputs.YPR
/ Appendix A / BYOB and Video Files / Rocket Maze / Rocket Maze 3 – Multiple Inputs.YPR
/ Appendix A / BYOB and Video Files / Rocket Maze / Rocket Maze 4 – Turtle Talk.YPR
/ Appendix A / BYOB and Video Files / Rocket Maze / Rocket Maze 5 – Loops.YPR
/ Appendix A / BYOB and Video Files / Rocket Maze / Rocket Maze 6 – Procedures.YPR
/ Appendix A / BYOB and Video Files / Rocket Maze / Rocket Maze 7 – Multiple Procedures.YPR
/ Appendix A / BYOB and Video Files / Rocket Maze / Video.MOV
/ Appendix A / BYOB and Video Files / Sentence Generator / Sentence Generator.YPR
/ Appendix A / BYOB and Video Files / Sentence Generator / Video.MOV
/ Appendix A / BYOB and Video Files / Traffic Light Simulator / Traffic Light Simulator.YPR
/ Appendix A / BYOB and Video Files / Traffic Light Simulator / Video.MOV
/ Appendix A / BYOB and Video Files / Twinkle Little Star / Twinkle Little Star.YPR
/ Appendix A / BYOB and Video Files / Twinkle Little Star / Video.MOV
/ Appendix A / BYOB and Video Files / Musical Pictures / Musical Pictures.YPR
/ Appendix A / BYOB and Video Files / Musical Pictures / Video.MOV
/ Appendix A / BYOB and Video Files / Draw It / Level 1 - House.YPR
/ Appendix A / BYOB and Video Files / Draw It / Level 2 - Street.YPR
/ Appendix A / BYOB and Video Files / Draw It / Video.MOV

Appendix B: Small Basic Microworld Programming Code and Data File for WP3 and WP4

Description

There is a PDF file containing the code that was used to create the Small Basic microworld activity. The .SB data file for viewing the Small Basic microworld activity can be opened by downloading *Small Basic* from http://smallbasic.com. There is also a video demonstration of the activity.

Directory/Filename

/ Appendix B / Small Basic microworld code.PDF
/ Appendix B / Small Basic microworld data file.SB
/ Appendix B / Video.MOV

Appendix C: Ethical Consent Forms

Description

This Appendix consists of three PDF files that were used to secure ethical consent. Two separate consent forms were used for (i) subject specialist teachers and (ii) learners/guardians in WP2, WP3 and WP4. A third form was used for learners and guardians in WP5.

Directory/Filename

/ Appendix C / Teacher Consent WP2, WP3, WP4.PDF
/ Appendix C / Learner and Guardian Consent WP2, WP3, WP4.PDF
/ Appendix C / Learner and Guardian Consent WP5.PDF

Appendix C1 – Learner and Parent/Carer Consent WP2, WP3, WP4

Ethical Consent For University of South Wales Study

What is the project?

- I want to find out if using certain computer-based activities helps children perform better in literacy.
- I also want to find out if using certain computer-based activities helps children perform better in computational thinking.
- I will compare work and look for differences and similarities between children who have completed the activities on the computer and those who have not.
- I will then turn this information into guidance for other teachers to help them with how they teach at the school. I may publish these findings to help teachers in other schools.

What are we asking you to do?

- Your school is part of the project which means they will be teaching some of the computer-based activities I have co-designed with them.
- These activities will form part of normal lessons and are about the normal things children would learn about in the curriculum.
- The school will share the children's work with me, so I can use it in my research. I will not collect the children's names though so I won't know whose work it is.
- I want to check that you are happy for me to use this work in my research project in this way.
- The university will keep the work for no more than three years and then destroy it, but the school can keep copies.

If you are happy for me to use the work in this way, you don't need to do anything.

If you would like more information before you decide, then please contact the link teacher in your school or Mr C Jenkins (Teacher of ICT and part-time research student) at craig.jenkins@southwales.ac.uk.

If you do not want work included in the project, that's fine. Just complete the opt-out form below and return it to the class teacher. Your student will still take part in the activities and not be disadvantaged in any way: their results will simply not be used in my research.

Research OPT-OUT response

I ………………………………………………. [child's name] do not want my work made available to the university.

Signature: Date:

And / or

I ………………………………………………. [parent / guardian name] do not want the work of

………………………………………………. [child's name] made available to the university.

Signature: Date:

Please return this form to the class teacher to ensure you are not included in the project.

Appendix C2 – Teacher Consent WP2, WP3, WP4

Ethical Consent For University of South Wales Study

What is the project?

- The project is part of a PhD research project by Craig Jenkins (Visiting Lecturer in ITT – University of South Wales; Teacher of ICT – Caerleon Comprehensive School).
- The project aims to establish whether a microworlds-based teaching scheme can bring about an improvement in academic performance in defined aspects of literacy.
- The project also examines whether such an approach has an impact upon computational thinking skills.
- It is hoped that this research will help form a series of effective practice recommendations for teachers wishing to replicate a microworlds-based approach to learning in their own classrooms.
- In order to achieve these aims, I want to analyse children's work by identifying the characteristics of their responses to some computer and paper-based tasks designed in conjunction with participating teachers.
- I may publish the findings through relevant teacher websites and research publications.

What am I asking teachers to do?

- If you agree to join the project, I would like to visit you to plan the classroom tasks at a time this is mutually convenient.
- There are then three stages to the project that will be led by you:
    - *Stage one:* You will need to complete a series of pre-intervention literacy **and** computational thinking exercises with **both** groups. No assessment is required: I will carry this out after agreeing a mark scheme with you. The final version that we agree upon will be emailed to you.
    - *Stage two:* Over a two-to-three lesson period, I would like you to implement online microworld-based activities with **one** of the groups only. I would like to observe some of these lessons in a technical support capacity and I will be available to support delivery of the teaching scheme. You will need access to a bookable computer suite at your school for this.
    - *Stage three:* You will need to complete a series of post-intervention computational thinking **and** literacy exercises with **both** groups. Again, no assessment is required: I will carry this out after agreeing a mark scheme with you. The final version that we agree upon will be emailed to you.
- Participating teachers will also have to administer an ethics process in their school, distributing letters to students and their parents / guardians to give individuals the opportunity not to share their work with the university.
- Whilst you can always withdraw from the project, I would urge you to discuss this with your Head Teacher / Principal before agreeing to participate to ensure there is support for you to commit to both meetings. I will also make contact with the Senior Leadership Team/Child Protection Officer at your school should you agree to participate.

Outcomes

- The minimum outcome will be a PhD thesis at the University of South Wales library. A copy of the findings will also be provided to you.
- I also intend to share my results through academic conferences and journals. It is hoped that this will help other educators wishing to replicate a microworlds-based learning approach in their classrooms.

Other important information

- I will report all findings anonymously - children's work will appear with a code or pseudonym.
- If you and the school Principal agree I will keep the identity of the school anonymous.
- If you agree to participate now you can withdraw at any point in the project without reason.
- As the activities used in class will be co-designed by you and have an educational purpose, I do not believe children will be disadvantaged in any way from this process. I will however prepare an information sheet for you to share with children and parents / guardians providing them with an opportunity to opt out of the research. Those opting out may participate in the class as an educational activity but their work should not be submitted as data for analysis.
- Student work will be kept by the researcher and destroyed after a period of up to three years, although the school may keep copies of the work.
- I will not be able to offer financial compensation for your involvement in this project.

Consent Response

□ I agree to participate in the project, although I understand I may withdraw at a later date
□ I have support from my Principal / Head Teacher
□ I agree to administer the student / parent consent process in my school

Teacher signature:

School:

Age group of students likely to be involved:

Date:

Appendix C3 – Learner and Parent/Carer Consent WP5

Ethical Consent For University of South Wales Study

What is the project?
I am currently participating in research activity with the University of South Wales that aims to explore the learning that can take place through the act of programming a computer. The project aims to find out if learning how to code, using particular computer-based activities, has the potential to develop children's literacy and computational thinking skills.

Key information

- We will be piloting some new coding activities during a series of upcoming lunchtime coding club sessions, that your child has volunteered for and actively expressed an interest in.
- I want to check that you are happy for me to look at the work they complete during this time.
- As part of the research, your child – as long as they are comfortable with this - may also be anonymously audio recorded to find out what their thoughts and views are when undertaking the coding activities that they are trialling.
- Where your child agrees to provide an audio recording of their views, all audio files will be destroyed as soon as they are transcribed into written form.
- Absolutely no personal data will be retained throughout this process.
- Returning this opt out form does not prevent your child from attending the coding club. It simply means that their work and views will not be passed on to the university.

What to do next

If you are happy with the above, you do not need to do anything further.

If you **do not** want your child's work or views to be included in the project, that's fine. Just complete the **opt-out form** below and return it to me as soon as possible.

If you would like more information before you decide, then please contact Mr C Jenkins at craig.jenkins@southwales.ac.uk.

Research OPT-OUT response - Child

I ………………………………………………. [child's name] do not want my work or views to be made available to the University of South Wales.

Signature: Date:

And / or - Research OPT-OUT response – Parent / Guardian

I ………………………………………………. [parent / guardian name] do not want the work or views of

………………………………………………. [child's name] made available to the University of South Wales.

Signature: Date:

Please return this form to Mr C Jenkins to ensure that your work and views are not passed on to the university.

Appendix D: Work Package 2 Intervention

Description

Appendix D consists of files relating to the intervention that was carried out in WP2.

A PDF file is also included in the root folder that contains an instructions sheet for teachers alongside a video demonstration of the activity.

A subfolder in this Appendix called *Snap! Files* includes two XML files of the microworld that are (i) a blank microworld and (ii) a fully-populated example. These can be opened by running Snap! in the web browser at http://snap.berkeley.edu/run.

A subfolder called *SPSS* contains files related to the quantitative analysis. The SPSS dataset used is included here. There is also a PDF file showing the tabular outputs generated by the analyses.

A final subfolder called *tests* is where pre-tests and post-tests for the intervention can be viewed.

Directory/Filename

/ Appendix D / Teacher instruction sheet.PDF
/ Appendix D / Video.MOV
/ Appendix D / Snap Files / Snap Microworld (Blank).XML
/ Appendix D / Snap Files / Snap Microworld (Exemplar).XML
/ Appendix D / SPSS / SPSS Data File.SAV
/ Appendix D / SPSS / Tabular outputs.PDF
/ Appendix D / Tests / Computational thinking tests.PDF
/ Appendix D / Tests / Literacy tests.PDF

University of South Wales
Prifysgol De Cymru

Newport School of Education
University of South Wales
Lodge Road
Caerleon NP18 3QT

Microworlds Research Project
Online Activity Information Sheet

Suggested Time for Activity: 2 – 3 lessons

1. Loading the microworld activity

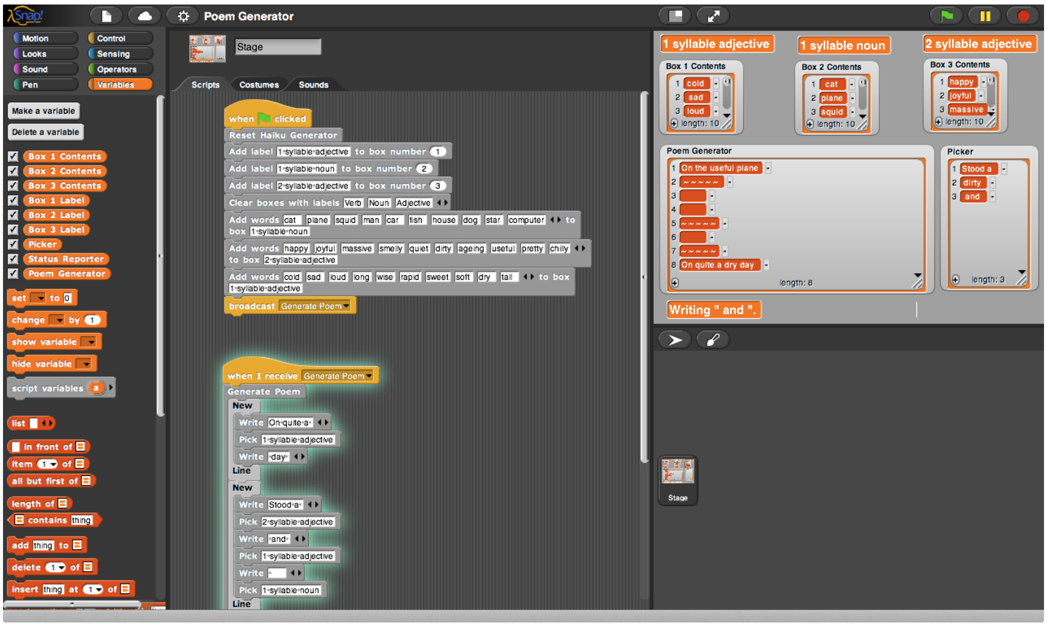
Step one: Add the attached *Microworld Loading File.XMS* to a shared network drive.

Step two: Visit http://snap.berkeley.edu/run in a web browser

Step three: In the File menu (indicated by a page of paper in the top-left) choose 'import' and navigate to the *Microworld Loading File.XMS* file on the network drive.

You are now ready to begin.

2. Using the Activity



```
when [flag] clicked
Reset Haiku Generator
Add label 1 syllable adjective to box number 1
Add label 1 syllable noun to box number 2
Add label 2 syllable adjective to box number 3
Clear boxes with labels Verb Noun Adjective
Add words cat plane squid man car fish house dog star computer to box 1 syllable noun
Add words happy joyful massive smelly quiet dirty ageing useful pretty chilly to box 2 syllable adjective
Add words cold sad loud long wise rapid sweet soft dry tall to box 1 syllable adjective
broadcast Generate Poem

when I receive Generate Poem
Generate Poem
  New
    Write On quite a
    Pick 1 syllable adjective
    Write day
  Line
  New
    Write Stood a
    Pick 2 syllable adjective
    Write and
    Pick 1 syllable adjective
    Write
    Pick 1 syllable noun
  Line
```

The stage area of this activity (top right) is comprised of four list monitors: *Box 1*, *Box 2*, *Box 3* and *Poem Generator*. The first phase of the activity involves deciding what grammar component each box will represent and then populating these boxes with examples of that grammar. Learners are able to change the name of each box (a list) to a particular grammar component in the scripting pane (middle). In the screen grab, box 1 has been named ‘1 syllable adjective’ using a *label* block. Learners can place items into each box using the *add words* block. In the example, the item ‘cold’ has been added to the ‘1 syllable adjective’ box.

The second part of the activity involves the computer randomly generating lines, and indeed poems, according to the poetic structures that the learners have programmed. Learners place blocks within the *Generate Poem* C-shaped block, which has a *for times* variable to allow for iteration. Learners write individual words with the *write* block and randomly select words from the boxes using the *pick* block. The computer generates lines and entire poems depending on the sequence and input variables specified in the *write* and *pick* blocks. Learners, in turn, are able to observe both the regularities and irregularities of written English.

To add more of the grey blocks to the scripting pane, go to the blocks pane in the far right and choose the *Variables* category from the top. When you scroll to the bottom of the variables block chooser, the grey boxes needed for the task are displayed.

Appendix D2 – Abridged WP2 computational thinking test questions

Exercise C1: Abstraction

Abstraction is used when programming a computer to create a solution that can solve many problems. In English, we can also create abstract sentences by looking for patterns in the way sentences are made up.

Write the abstract sentences for the sentence pairs below by identifying the articles, verbs, nouns and adjectives. The first one has been done for you.

Example

Sentence 1: The big, red dog.
Sentence 2: A screaming, loud baby.

Abstract Sentence: **Article adjective , adjective , noun .**

Question 1 *(__ / 3 marks)*

Sentence 1: A swimming fish.
Sentence 2: The running man.

Abstract Sentence: ....

Question 2 *(__ / 3 marks)*

Sentence 1: A deafening bang.
Sentence 2: An angry monster.

Abstract Sentence: ....

Question 3 *(__ / 5 marks)*

Sentence 1: The scary lion was roaring.
Sentence 2: The happy child was laughing.

Abstract Sentence: ....

Unabridged tests appear on the data disc.

Appendix D3 – Abridged WP2 literacy test questions

Exercise P4: Poetry forms

You know already that the *haiku* uses a fixed number of syllables over three lines in the form **5-7-5.** Some others forms of poetry use different numbers of syllables. A tanka uses the form **5-7-5-7-7** and a cinquain uses **2-4-6-8-2**.

Cross out some words in the poems below to make them fit the correct form of the poem specified. One has been done for you.

Example

Haiku

Spring is in the air
*Flowers are blooming sky high ~~**in the sun**~~*
*Children are ~~**playing and**~~ laughing*

Question 1 *(__ / 3 marks)*

Cinquain

Gazing,
At the bright moonlight,
And entranced by its beauty,
Pondering the possibilities,
Quiet,

Question 2 *(__ / 5 marks)*

Tanka

I love my cheeky kitten,
She is so little and cute,
She has a pink tongue,
And lots of long whiskers too,
She purrs when I stroke her furry back,
She purrs loudly.

Question 3 *(__ / 2 marks)*

Haiku

I walk across sand quickly,
And I find myself blistering,
In the hot, hot heat.

Unabridged tests appear on the data disc.

Appendix E: Work Package 3 Intervention

Description

Appendix E consists of files relating to the intervention that was carried out in WP3.

A subfolder called *SPSS* contains the SPSS dataset and syntax calculations that were used as part of the analysis. The formatted outputs are available in a PDF file.

The subfolders called *Tests* contains the pre-tests and post-tests for literacy and computational thinking.

Directory/Filename

/ Appendix E / SPSS / SPSS Data File.SAV
/ Appendix E / SPSS / SPSS Calculations Syntax.SPS
/ Appendix E / SPSS / Tabular Outputs.PDF
/ Appendix E / Tests / Computational Thinking Post-tests.PDF
/ Appendix E / Tests / Computational Thinking Pre-tests.PDF
/ Appendix E / Tests / Literacy Post-tests.PDF
/ Appendix E / Tests / Literacy Pre-tests.PDF

Appendix E1 – Abridged WP3 computational thinking test questions

Question 1 Find the **coding errors** and write the corrections.

(7)

```
1 Start Poem
2         Repeat (Syllable, 5)
3     NewLine
4          Repeat Syllable, 7)
5     New Line
6          Repeat (Syllabel, 5)
7 EndPoem
```

(1) Line Number: ....3... ✏ Correction: .... ✏

(2) Line Number: .... ✏ Correction: .... ✏

(3) Line Number: .... ✏ Correction: .... ✏

(4) Line Number: .... ✏ Correction: .... ✏

Question 2 Complete the **sequence** order to write a **cinquain.**

(10)

```
1 Start Poem
~~2 EndPoem~~
3     New Line
4     New Line
5     New Line
6     New Line
7         Repeat (Syllable, 2)
8         Repeat (Syllable, 2)
9         Repeat (Syllable, 4)
10        Repeat (Syllable, 6)
11        Repeat (Syllable, 8)
```

(1) Line Number: .... ✏ (2) Line Number: .... ✏

(3) Line Number: .... ✏ (4) Line Number: .... ✏

(5) Line Number: .... ✏ (6) Line Number: .... ✏

(7) Line Number: .... ✏ (8) Line Number: .... ✏

(9) Line Number: .... ✏ (10) Line Number: .... ✏

(11) Line Number: .....2... ✏

Unabridged tests appear on the data disc.

Appendix E2 – Abridged WP3 literacy test questions

Question 11 Write down the **word types.**

(3)

They (1) loudly (2) screeched then (3) flew away.

(1) .... ✏ (2) .... ✏ (3) .... ✏

Question 12 Write words of the correct **type** and **number of syllables.**

(3)

Examples of **connectives** with **one syllable.**

(1) .... ✏ (2) .... ✏ (3) .... ✏

Question 13 In the space below, write your own **tanka**.

(10)

.... ✏

.... ✏

.... ✏

.... ✏

.... ✏

.... ✏

.... ✏

.... ✏

Unabridged tests appear on the data disc.

Appendix F: Work Package 4 Intervention

Description

Appendix F consists of files relating to the intervention that was carried out in WP4.

A subfolder called *SPSS* once again contains the SPSS dataset, syntax calculations and the tabular outputs generated as a result of the quantitative analyses.

The subfolder called *Teaching and Learning Resources* contains five files:

1. A set of supporting PowerPoint slides that functioned as a teaching aid for introducing the concepts and practices of computational thinking.

2. A second set of slides that were used when introducing the aspects of literacy.

3. A teaching aid to give learners more guidance when using the text-based Small Basic activity when in the ICT suite.

4. A set of practice worksheets in computational thinking that were distributed to learners for the 'unplugged' activities at the beginning of the intervention.

5. A set of literacy practice worksheets.

Directory/Filename

/ Appendix F / SPSS / SPSS Data File.SAV
/ Appendix F / SPSS / SPSS Calculations Syntax.SPS
/ Appendix F / SPSS / Tabular Outputs.PDF
/ Appendix F / Teaching and Learning Resources / Computational Thinking Slides.PDF
/ Appendix F / Teaching and Learning Resources / Computational Thinking Practice Worksheets.PDF
/ Appendix F / Teaching and Learning Resources / Literacy Slides.PDF
/ Appendix F / Teaching and Learning Resources / Literacy Practice Worksheets.PDF
/ Appendix F / Teaching and Learning Resources / Small Basic Slides.PDF

Exercise C2: Sequencing

Sequencing when programming a computer is about writing a series of steps or instructions for the computer to follow. Many poems in English make use of sequencing.

A robot wants to write a poem containing three lines made up of two, four and six syllables. Below is an example of the poem.

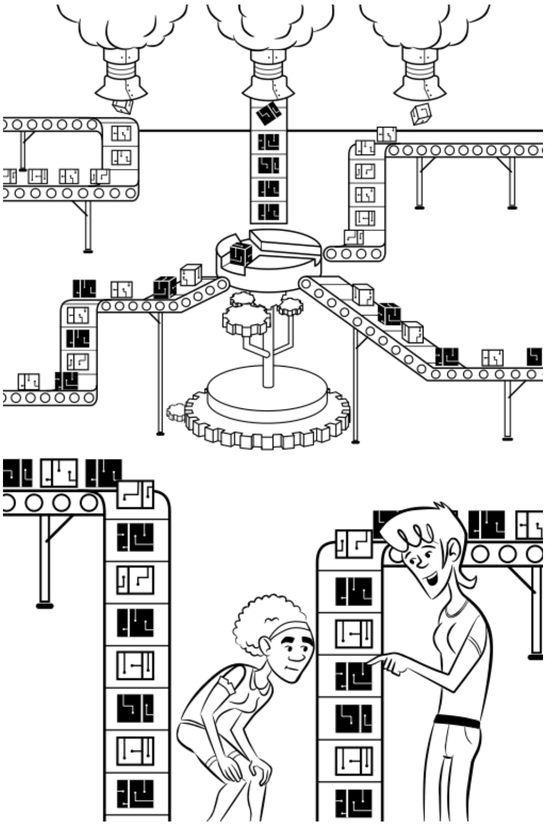
Loudly
The tree did rock
On the loud and dark night

To write this poem, the robot carries out the following three commands.
Repeat (*Instruction, Number of times* **)** tells the robot to repeat what is in the bracket for a given number of times.
New Line tells the robot to start a new line on the page.
Syllable tells the robot to write a word with one syllable.

The robot has started writing the first line of his poem. Can you program him to write the last two lines of the poem?

Question 1 *(__ / 7 marks)*

```
Start Poem

            Repeat (Syllable, 2)

    New Line


..............................................................


..............................................................


..............................................................


End Poem
```



Unabridged activities appear on the data disc.

Appendix F2 – Abridged WP4 'unplugged' literacy activities

Exercise P5: Poetry Analysis

1 Listen…
2 With faint dry sound,
3 Like steps of passing ghosts,
4 The leaves, frost crisp'd break from the trees
5 And fall.

Look at the extract from the poem *November Night* by Craspey and answer the questions that follow. One has been done for you.

Example

State the **number of syllables** in line **six**. Four

Question 1 *(__ / 1 marks)*

What **fixed poetry form** is this? ………………………………

Question 2 *(__ / 4 marks)*

Write the **number of syllables** in each line for lines **2-5**). ………………………………

Question 3 *(__ / 4 marks)*

Give four **stressed** syllables that are used in line **4.** ……………………… ………………………

……………………… ………………………

Question 4 *(__ / 8 marks)*

Write down your own poem, in the format of a **tanaga,** which is based on some of the ideas in the poem you have read.

………

………

………

………

………

………

Unabridged activities appear on the data disc.

Appendix G: Work Package 5 Extracurricular Club

Description

This Appendix mirrors the structure of the WP and is split into two subfolders for phase 1 and phase 2.

All of the tutorials used for phase 1 can be opened by running Snap! in the web browser at http://snap.berkeley.edu/run. There is also a video demonstration providing an example of one the tutorials

The Phase 2 folder comprises a video demonstration and a further three subfolders:

1. A full breakdown of the CA frequencies recorded during the SEDA analysis.
2. A blank and exemplar XML file that can be used to import the microworld activities into Snap.
3. An instructions booklet and task brief that was given to learners during the Tanaga Challenge.

Directory/Filename

/ Appendix G / Phase 1 / Video Demonstration.MOV
/ Appendix G / Phase 1 / Snap Files / Tutorial 1 - The Green Flag.XML
/ Appendix G / Phase 1 / Snap Files / Tutorial 2 - The Write Block.XML
/ Appendix G / Phase 1 / Snap Files / Tutorial 3 - The Pick Random Word Block.XML
/ Appendix G / Phase 1 / Snap Files / Tutorial 4 - The New Line Block.XML
/ Appendix G / Phase 1 / Snap Files / Tutorial 5 - The Pick Number Block.XML
/ Appendix G / Phase 1 / Snap Files / Tutorial 6 - The Generate Poem Block.XML
/ Appendix G / Phase 1 / Snap Files / Tutorial 7 - The Clear Poem Generator Block.XML
/ Appendix G / Phase 1 / Snap Files / Tutorial 8 - The Clear All Block.XML
/ Appendix G / Phase 1 / Snap Files / Tutorial 9 - The Give Name Block.XML
/ Appendix G / Phase 1 / Snap Files / Tutorial 10 - The Add Words Block.XML
/ Appendix G / Phase 1 / Snap Files / Tutorial 11 - The Clear Word Number Block.XML
/ Appendix G / Phase 1 / Snap Files / Tutorial 12 - Getting Started.XML
/ Appendix G / Phase 2 / Video Demonstration.MOV
/ Appendix G / Phase 2 / SEDA Data / CA Frequencies.PDF
/ Appendix G / Phase 2 / Snap Files / Tanaga Challenge - Blank.XML
/ Appendix G / Phase 2 / Snap Files / Tanaga Challenge - Example.XML
/ Appendix G / Phase 2 / Teaching and Learning Resources / Challenge Brief.PDF
/ Appendix G / Phase 2 / Teaching and Learning Resources / Instructions.PDF

Appendix G1 – SEDA scheme CA frequencies

CA Frequencies (SEDA Scheme) for WP5

		CA Frequencies	
		Teachers	**Learners**
	I – Invite elaboration or reasoning	**16**	**9**
I1	Ask for explanation or justification of another's contribution		
I2	Invite building on / elaboration / (dis)agreement / evaluation of another's contribution or view		
I3	Invite possibility thinking based on another's contribution		
I4	Ask for explanation or justification	15	9
I5	Invite possibility thinking or prediction		
I6	Ask for elaboration or clarification	1	
	P – Positioning and Coordination	**0**	**2**
P1	Synthesise ideas		
P2	Evaluate alternative views		
P3	Propose resolution		1
P4	Acknowledge shift of position		
P5	Challenge viewpoint		1
P6	State (dis)agreement/ position		
	RD – Reflect on dialogue or activity	**0**	**1**
RD1	Talk about talk		
RD2	Reflect on learning process/ purpose/ value/ outcome		1
RD3	Invite reflection about process/ purpose/ value/ outcome of learning		
	E – Express or invite ideas	0	5
E1	Invite opinions/beliefs/ ideas		
E2	Make other relevant contribution		5

		CA Frequencies	
		Teachers	**Learners**
	R – Make reasoning explicit	**0**	**20**
R1	Explain or justify another's contribution		2
R2	Explain or justify own contribution		8
R3	Speculate or predict on the basis of another's contribution		4
R4	Speculate or predict		6
	B – Build on ideas	**0**	**4**
B1	Build on /clarify others' contributions		
B2	Clarify/elaborate own contribution		4
	C – Connect	**1**	**3**
C1	Refer back	1	2
C2	Make learning trajectory explicit		
C3	Link learning to wider contexts		1
C4	Invite inquiry beyond the lesson		
	G – Guide direction of dialogue or activity	**18**	**0**
G1	Encourage student-student dialogue		
G2	Propose action or inquiry activity	14	
G3	Introduce authoritative perspective		
G4	Provide informative feedback	1	
G5	Focusing		
G6	Allow thinking time *[optional when not verbally explicit]*	3	

Appendix G2 – Abridged WP5 activity brief

Clwb Côd | Code Club

Tanaga Competition Brief

What is a tanaga?

The tanaga is a type of short Filipino poem, consisting of four lines with seven syllables each with the same rhyme at the end of each line --- that is to say a 7-7-7-7 syllable form, with an AAAA rhyme pattern.

See waves foaming, wash the shore.
The castle is there no more.
No windows remain; no door.
Roof of sky; a sandy floor.

'Tanaga' by Benjamin Garrett
Winner of the allpoetry.com tanaga competition

Competition rules

Over the next few sessions at *Clwb Côd*, you will create your own tanagas using the poem generator microworld we have been using. At the end of this challenge, we will present our poems to each other.

Prizes are available for the winning entries.

The winner will be chosen according to the following criteria:

1. How **creative** your poem is

2. Whether or not your poem meets the structure of syllables and rhymes for a tanaga poem

3. The features of the poem generator microworld that you have used to create your poem.

Unabridged brief appears on the data disc.

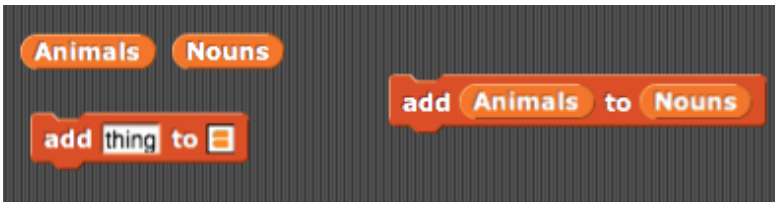
Workshee 7

Boxes within boxes

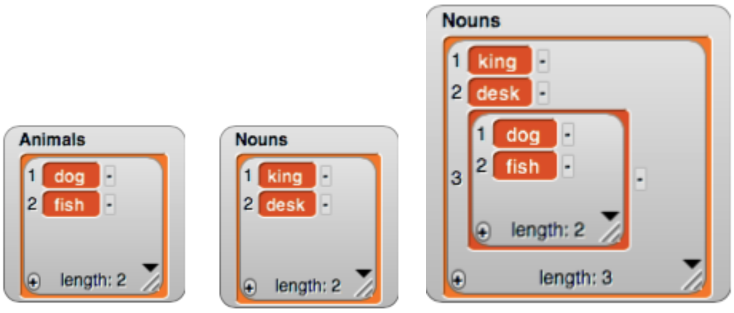
Step one: boxes within boxes

Lets say that you have a box called ‘animals’ and a box called ‘nouns’. It may make sense to combine these boxes into one bigger box called ‘nouns’. This is how we would do it.

The rounded box name blocks can be placed within areas where you can type as well as on top of box icons within blocks. This makes it easy to place boxes within boxes. See the images below to see how this happens.

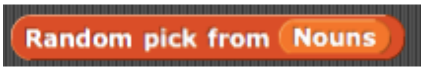


```
Animals   Nouns

add thing to ☰

add Animals to Nouns
```



Step two: the random pick from block

If the computer was to randomly pick item 3 from the nouns box above, currently it would bring back the entire list of words for this ‘box within a box’ - dogfish. We will need to use a special block for picks from these special boxes with other boxes inside them. The **random pick from** block will automatically work out for us whether the item that has been chosen is a box within a box. If it is, it will only bring back one of these words rather than all of them!



```
Random pick from Nouns
```

Unabridged resources appear on the data disc.

Appendix H: Project outline diagram

Description

This outline diagram provides a summary of the methods, school contexts, participants, platforms, curriculum contexts and the planning-intervention-evaluation cycle across the five work packages. Included on the second page are the key quantitative findings in addition to guidelines for their interpretation.

Directory/Filename

/ Appendix H / Project outline diagram.PDF

Appendix H1 – Project outline diagram

Thesis on a page

	Work Package 1	Work Package 2	Work Package 3	Work Package 4	Work Package 5
School Context					
School 1	🟨		🟨	🟨	🟨
School 2		🟨			
Participants					
Year 7	🟨		🟨	🟨	🟨 Mixed
Year 8		🟨			
Mixed ability	🟨		🟨	🟨	🟨
Low ability		🟨			
Platforms for the intervention activities					
BYOB	🟨				
Small Basic			🟨	🟨	
Snap!		🟨	🟨	🟨	🟨
Curriculum Context (* = focus on cross-curricular literacy)					
ICT#	🟨				
English*		🟨			
Drama*			🟨	🟨	
Extracurricular club*					🟨
Action research methodology – Evaluation informed planning					
	• Pilot • Block based programming across subject disciplines	• Finalised guiding questions. • Focus on literacy for second guiding question.	• More robust quantitative tests. • Re-test for link between CT and literacy. • Triangulate with qualitative data.	• Greater controls on test timings. • More ‘unplugged’ activities. • Pair programming • Exhibition format at end of scheme.	• Incremental approach – tutorial series. • Dialogic teaching. • Extended study in extracurricular setting.

Thesis on a page

	Work Package 1	Work Package 2		Work Package 3		Work Package 4		Work Package 5
Qualitative analyses								
Survey/self-reporting	🟨			🟨				
Dialogue analysis *(Wegerif & Mercer)*						🟨		
Dialogue analysis *(SEDA)*								🟨
Quantiative analyses								
		Lit.	***CT***	***Lit.***	***CT***	***Lit.***	***CT.***	
T-test p value		0.242	0.202	0.029	0.018	0.092	0.012	
Cohen's d effect size		0.46	0.50	0.63	0.68	0.54	0.83	
Pearson correlation (CT = predictor)		+0.341						
F-test (CT = predictor)		0.082						
MLR (CT + grouping = combined predictors) Adjusted R^2		0.068		0.141				

Quantitative Guidelines (Muijs, 2011) – Colour coding added for clarity

T-tests / F-tests	Cohen's D Effect Size	Pearson's R Correlation	Adjusted R^2
<0.05 = Significant	<1.00 = strong effect	<+/-.8 = strong correlation	> 0.5 = strong fit
>0.05 = Not significant	0.51 – 1.00 = moderate effect	<+/-.5 = moderate correlation	0.31 – 0.5 = moderate fit
	0.21 – 0.50 = modest effect	<+/-.3 = modest correlation	0.11 – 0.3 = modest fit
	0 - 0.20 = weak effect	<+/-.1 = weak correlation	<0.1 = poor fit