CONSTRUCTING PROGRAMS – HOW CHILDREN WITH ATTENTION DEFICIT HYPERACTIVITY DISORDER (ADHD) LEARN TO PROGRAM

by

COLIN LEON PILKINGTON

submitted in part fulfilment of the requirements for
the degree of

MASTER OF SCIENCE IN MATHEMATICS, SCIENCE,
AND TECHNOLOGY EDUCATION

in the subject

COMPUTING EDUCATION

at the

UNIVERSITY OF SOUTH AFRICA

SUPERVISOR: MS JH GELDERBLOM

NOVEMBER 2007

Summary

Many learners find the study of introductory computer programming difficult. This is also true of children with attention deficit hyperactivity disorder (ADHD), and we need an improved understanding of how they learn programming. After reviewing the constructivist approach to teaching and learning and investigating ADHD, this study explored strategies for constructive learning of introductory programming. The aim was to evaluate the effectiveness of the Karplus learning cycle to teach introductory programming. This was done through qualitative research from an interpretive perspective. Action research techniques were employed and data analysed using grounded theory methods. Four major constructivist teaching categories emerged, all of which support the use of the Karplus cycle. It is concluded that the three-phase Karplus cycle can be used to assist these learners learn introductory programming. However, it needs to be understood more broadly and the middle phase broken into two subphases to ensure effective learning.

Key terms

attention deficit hyperactivity disorder; ADHD; computer programming; introductory programming; Karplus learning cycle; constructivism; grounded action research

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1 Introduction

> When interest goes beyond the individual classroom, to examine the efficacy of specific approaches or techniques, to judge the generalisability or transferability of outcomes, to work to understand whether there are a set of conditions or abilities that pre-dispose for success in [computer science], then we move towards "Computer Science Education Research." (Fincher in Clancy, Stasko, Guzdial, Fincher & Dale, 2001, p. 337).

Learning computer programming is difficult and especially so for the novice programmer who comes to the topic with no effective model of the computer on which to base the concepts and structures needed to effectively learn programming (Ben-Ari, 1998; Robins, Rountree & Rountree, 2003). The question, then, is "[w]hat can we as teachers do to most effectively support novice programmers?" (Robins et al., 2003, p. 138). In this chapter, I will consider a background to learning computer programming as well as set out the boundaries of this study and its significance. A chapter overview will also be provided.

1.1 Background to the study

When it is accepted that many learners find the study of computer science extremely difficult, especially at introductory levels (Ben-Ari, 1998; Robins et al., 2003), then it is understandable that the teaching and learning of computer science need to improve. Further, there needs to be a better understanding of how learners learn computer science concepts (Ben-Ari, Berglund, Booth & Holmboe, 2004). This is true of children with attention deficit hyperactivity disorder (ADHD), too – we need to improve our understanding not just of how children with ADHD learn, but of how they learn computer science and programming in particular.

There are generally two different models of teaching and learning in current computer science classrooms (Van Gorp & Grissom, 2001). Objectivist classrooms tend to focus on the teacher. The teacher provides the information to be learnt, and the task of the learner is to listen to the teacher, practise the skills learnt to reinforce the learning, and be tested in a quantifiable manner. The constructivist classroom,

on the other hand, focuses on the learner. It is the learner's experiences that lead to learner constructions that provide the learner with meaning and understanding. Testing is then about being able to defend a position taken and use the information in an effective manner. The focus in this study will be on constructivist approaches to teaching and learning. However, the learners involved were not just any group of learners in general, but particularly those with ADHD.

One of the groups of underachieving learners that often causes despair among teachers and parents, and is receiving increasing attention at the moment, is learners with ADHD (Brand, Dunn & Greb, 2002; Burgess, 2003). Unfortunately, it is often the problems around the attention differences and hyperactivity of these learners that receive attention (Bester, 2000). However, these children can learn (Shore, 1998). The process of learning more about ADHD has included searching for approaches to help such children learn better and has even included calls for teachers to experiment with various teaching variables in an effort to advance knowledge about teaching children with ADHD (Brand et al., 2002). Such teaching is about another perspective on learning and encompasses different boundaries to those found in the average classroom (Bester, 2000).

Why teach computer programming at all? There is evidence in the USA that most learners who take computer science at school will never write another computer program (Clement, 2004). Thus, in this established and constantly growing field of computer science education research (Clancy et al., 2001; Pears, Daniels & Berglund, 2002), the focus has often switched away from goals aimed at preparing learners for a programming career to those aimed at improving and growing thinking and problem-solving ability (Clement, 2004; Jakovljevic, Ankiewicz & De Swardt, 2003). If the aim is then to learn more about the mental processes involved in problem-solving situations within the context of computer programming, qualitative research approaches are more useful than quantitative approaches (Hazzan, Dubinsky, Eidelman, Sakhnini & Teif, 2006). Therefore, this study will take such an approach in examining an introductory computer programming course.

1.2 Rationale

Lui, Kwon, Poon, and Cheung (2004) note that although there is an abundance of literature on better ways to teach introductory programming, there is virtually nothing for helping weak learners, or learners with special educational needs, learn programming. Lui et al. (2004) further believe that constructivism, with its focus on the learner rather than the content, may offer a promising starting point.

A teacher's primary responsibility is the learner's learning (Feldman & Minstrell, 2000). Given that teachers are often expected to change how they teach (although they are seldom given any meaningful data that encourages this change (Clement, 2004)), this study aims at a better understanding of ADHD learners and their learning and, thus, provides another approach in the continued efforts to develop teaching (Ben-Ari et al., 2004).

This study will focus on constructivist approaches to the teaching and learning of computer programming. Thus, the approach taken will consciously move away from that taken in most introductory programming courses where lectures are used to introduce topics and ideas, followed by laboratory exercises to reinforce the concepts (Kölling & Barnes, 2004). Current textbooks tend also to begin with definitions and are ordered according to programming language constructs, with several small examples (Clement, 2004; Kölling & Barnes, 2004; Robins et al., 2003). Rather, the Karplus learning cycle, which Karplus (1980) suggested as an approach to teaching for the development of reasoning, will be used to structure learning in the classroom. The cycle's three phases – exploration, concept introduction, and concept application – will be used to give learners a chance to see a concept in action, to learn about it, and to put it into practice. This study would, thus, be in keeping with the current constructivist focus in technology classrooms where there is a move towards developing thinking and problem-solving skills (Jakovljevic et al., 2003; Van Gorp & Grissom, 2001).

The teacher's task is "not to construct a new horizon for each student ... . Instead she needs to seek a way to merge horizons – for her students' horizons to fuse with

hers so that they together begin to see the world in the way that she envisions it" (Feldman, 1994, p. 96). Researching the teacher's task, then, is about evaluating the teacher's decisions and actions and how these include the learners and lead to new, shared educational situations, remembering that teachers are concerned about the level of their effectiveness (Abell, 2005; Feldman, 1994). The relationship of teacher and evaluator is then a dialogical one through discourse with educational situations in the past and present, aiming at better understanding of, and improved, practice (Feldman, 1994).

Thus, the task of research is not to prove causality or find generalisable principles, but to find ways in which teachers can create new educational environments and approaches that include learners in a way that leads to a merging of horizons (Feldman, 1994) and so improve teaching practice. Feldman (1994) points out that an assessment of what learners have learnt, how their attitudes have changed, or their new ways of thinking will not lead to an understanding of educational situations that results in the merging of horizons. Rather, careful attention should be paid to educational practice, which can be used to shape educational situations aimed at merging horizons, and the achievement of learning goals.

This study is, however, not intended to result in a definitive teaching method for ADHD learners being introduced to computer programming. Rather, it aims at another approach, an alternative methodology, to constructing computer programs that can be used in conjunction with the array of approaches available to teachers in helping learners learn.

Benefits for me as teacher include my professional development in terms of creating educational environments that promote teaching and learning of computer programming for learners with ADHD. This experience can then lead to an educational approach that other teachers could use to structure their learning environments. This is a very practical or pragmatist result, providing practical lesson planning aids for teachers in terms of the Karplus cycle's particular lesson structure and approach.

However, it could also lead to theoretical modifications of constructivist models that guide computer programming education. Thus, the focus on a particular approach, the Karplus learning cycle in this case, will allow me to determine to what extent this cycle can be used and where modifications may add to its efficacy in helping learners learn.

1.3 Problem statement, thesis statement, and research questions

Constructivist teaching methods have often been applied to teaching introductory programming, but very little attention has been given to how learners with special educational needs, such as learners with ADHD, can be supported through constructivist methods to learn computer programming.

It will be argued that constructivist teaching methods, and the Karplus learning cycle in particular, can be used to effectively facilitate the learning of introductory computer programming by learners with ADHD.

Exploring the Karplus learning cycle should help gain some understanding of the extent to which constructivism does offer a starting point for reaching learners with special educational needs, and those with ADHD in particular, and whether constructivism does promote learning. The focus in this study will be on the Karplus cycle and its role in the structuring of lesson flow and whether it does lead to educational environments and practice that promote learning of computer programming.

The questions that will guide my research are:

1. What is constructivism, and how can it be used to support the teaching and learning of programming?
2. What is the Karplus learning cycle, and how can it be used to support the teaching and learning of computer programming?
3. What is ADHD, and what are the strategies that can be used to promote learning by children with ADHD?

4. Is the Karplus constructivist learning cycle successful in the teaching of computer programming to children with ADHD?

1.4 Research design and methodology

The research design is based on a framework provided by Pears et al. (2002) to ensure a rigorous study in the field of computer science education. An empirical qualitative study was undertaken, using an interpretive philosophical paradigm to guide the research. Action research techniques were employed to implement the study, with the researcher acting as teacher.

The research was carried out over two courses in introductory programming at a secondary school for children with special educational needs. The first course took place in 2006 and consisted of 11 lessons of one hour each in the course that covered the basics of programming from variables to `If` statements and `While` and `For` loops. The second course took place in 2007 and started with some basic techniques in Delphi, covering the use of forms and basic components (such as buttons, labels, and edit boxes). In this course, there were 12 lessons of one and a half hours each. The learners were Grade 9 to 12 boys, all displaying behaviour associated with ADHD.

Data was collected using observation, interviews, diary writing, and documents. The research material so gained was analysed using a version of grounded theory modified for use with action research. Grounded theory was used, as it has been found to be useful in developing context-based, process-oriented descriptions and explanations (Myers, 1997). The grounded theory analysis identified 22 concepts in the data, describing various activities and responses to the approaches attempted in the courses. Through further analysis, four major themes, or categories, emerged in the use of constructivist teaching approaches to the learning of introductory programming relating to the Karplus learning cycle, questioning technique, the time needed to learn, and planning and experimentation.

Thus, two of the consistently advocated research techniques of interpretivist research (action research and grounded theory) (De Villiers, 2005) were used in this study.

1.5 Boundaries and assumptions

This study will focus on the use of the Karplus learning cycle, and it will be its use that will receive most of the attention. Although other constructivist approaches were also used in the teaching of the introductory programming courses on which the study is based, their significance will be largely assumed, as they are fairly standard constructivist techniques. Thus, the interventions that were attempted in the courses centred on the structuring of the learning environments in terms of learning cycles.

Similarly, the effectiveness of the teaching approaches that are suggested for ADHD learners will also be assumed, as they, too, are highlighted by most authorities as techniques that have shown results. Though some of the responses to the learning environment may be as a result of ADHD behaviour, this will only be mentioned in passing and will not receive much attention in the assessment of the results of the teaching interventions attempted.

Further, the conclusions drawn are the result of a single action research cycle. The study's findings are, thus, limited in that the diagnosing and action planning that took place at the start of the second action research cycle are not reported here. The second action research cycle has not progressed far enough as yet to indicate whether the proposed changes to the Karplus learning cycle will have an effect on the effectiveness of the use of the cycle in learning introductory computer programming. The application of the results to similar learning environments may also be slightly compromised by the small numbers of learners on which this study was based.

In terms of the evaluation of the research in section 6.3, this study could have taken a more critical view of the classroom context. There was no examination of the historical context of the classroom and no exploration of the preconceptions that the

teacher had of learners with ADHD or of learning programming in general. Also, the principles of evaluation require that multiple interpretations of the participants be explored and that the researcher seek what is behind what is being said by participants. These two requirements should ideally be explored further.

However, I believe that these limitations do not invalidate what has been found in the study. These limitations do mean that the findings should be taken as preliminary until they are substantiated by further studies.

1.6 Chapter overview

The aim of the research is to examine the effectiveness of using constructivism (specifically the Karplus learning cycle) as an approach to teaching introductory computer programming to learners with ADHD.

Thus, an examination of the use of constructivism in computer programming in general will be undertaken (Chapter 2) as well as an examination of ADHD (Chapter 3) in an attempt to understand it and the various strategies that can be used to teach these particular learners. The research design and methodology used in the study will then be detailed following the framework suggested by Pears et al. (2002) (Chapter 4). The results of the study will be discussed (Chapter 5) and conclusions drawn (Chapter 6).

1.7 Conclusion

In this chapter, I provided an overview of this study that will investigate the effectiveness of constructivist methods in helping children with ADHD to learn computer programming. It is the Karplus learning cycle that will receive much of the attention in this study, but it is not the only factor at play in the context of the computer classroom in which the investigation was carried out. There are the other constructivist approaches to learning, in general, that supplement the learning cycle. Also, learners with ADHD behaviours are part of the classroom, too. It is to these two areas that the study turns first.

Chapter 2 Constructivism as an approach to learning programming

2.1 Introduction

Constructivism is a "philosophy about teaching and learning rather than a specific teaching method or approach" (Harris & Graham, 1996a, p. 135). Ben-Ari (1998) and Mayes (in Allen, 2005) point out that the dominant theory of learning currently is constructivism, where knowledge is actively constructed by learners rather than simply being passively absorbed from textbooks and lectures. Further, as constructivist classrooms are considered to be problem-solving environments and as computer science is about problem solving by nature (Van Gorp & Grissom, 2001), it seems reasonable to use constructivism in the teaching of computer science and, specifically, programming.

In this chapter, I will examine constructivism as an approach to teaching and learning, in general, and will highlight the main themes that can be found in the various forms of constructivism. The critiques of the more radical forms of constructivism will provide a more balanced view of the approach. As this study focuses on using constructivism as an approach to teaching introductory computer programming, I will explore the application of this theory of learning to programming specifically. Finally, the focus will shift to the use of learning cycles, and the Karplus learning cycle in particular, as the teaching on which this study is based was structured around the Karplus learning cycle.

Firstly, then, I will provide an overview of constructivism as an approach to learning and teaching.

2.2 What is constructivism?

Constructivism, a psychological and philosophical perspective on learning and understanding (Schunk, 2004), has been summarised by Bodner (1986, p. 873, italics in original) in a single statement thus: "*Knowledge is constructed in the mind of the learner*." Although constructivism is from the late 1980s and early 1990s,

aspects of it are not new (Van Gorp & Grissom, 2001), and it can be broadly categorised into two groups – radical and social constructivism – based on the relative importance of the individual and the group (Ben-Ari, 1998; Gijbels, Van de Watering, Dochy & Van den Bossche, 2006). Theorists such as Piaget would represent the role of the individual, and Vygotsky the role of society, in the construction of knowledge (Phillips, 1995). There is no one definition for constructivism (Harris & Graham, 1994), and there are many forms of, and different approaches to, its implementation, and, as such, constructivism can be seen to be an umbrella term for several perspectives on teaching and learning (Gijbels et al., 2006; Harris & Graham, 1994; Karagiorgi & Symeou, 2005; Perkins, 1999; Phillips, 1995). There are, however, several major principles that are common among most of the approaches based on constructivist thinking (Cey, 2001; Harris & Graham, 1994). The main points and basic assumptions that are common to most approaches are the following.

2.2.1 Constructing knowledge

The core notion of constructivism is that knowledge is not transmitted from teacher to learner, but built up by the learners themselves (Driver, Asoko, Leach, Mortimer & Scott, 1994; Karagiorgi & Symeou, 2005). The learner is not a passive recipient of knowledge, and knowledge is not gained by learners simply because information was transmitted from teacher to learner (Cey, 2001). Rather, the learner is an active participant, responding to experiences and the environment by constructing meaning and knowledge, which are then included in previously constructed knowledge (Harris & Graham, 1996a; Loyens, Rikers & Schmidt, 2006; Perkins, 1999; Powers, 2004; Von Glasersfeld, 1992). Constructivist approaches are, thus, learner-centred ones (Karagiorgi & Symeou, 2005).

2.2.2 Social context

Learning happens within social, meaningful, and authentic contexts (Harris & Graham, 1994). It is “coconstructed ... in dialogue with others” (Perkins, 1999, p. 7). The role of the social environment provides both a source of challenge to learners’ viewpoints and a source of models of what constructions should look like

(Wadsworth, 1996). The social interactions that mediate learning and development include parents, siblings, peers, teachers, and other significant people in a child's life (Flavell, Miller & Miller, 1993). The various people in the social environment can be divided into basic groups – adults and peers – and their roles in learning would differ slightly.

The role of the adults would be that of "guided participation" (Flavell et al., 1993, p. 16) – to guide, support, challenge, and model (Flavell et al., 1993). In a sense, the role of the adults is to help bridge Vygotsky's zone of proximal development (ZPD). The ZPD's lower limit is understood to be the level of problem-solving ability that a learner displays when working alone (that is, without any assistance) and its upper limit the additional problem-solving tasks that the learner can manage with the assistance of an instructor (Santrock, 2005). Thus, guided participation takes children from where they are and helps them to where they could be and could be seen as what Tharp and Gallimore (in Daniels, 2001, p. 59) refer to as "teaching as assisted performance".

The role of peers is to provide a challenge to the egocentrism of a learner and to provide a source of varying goals, multiple perspectives, and alternate viewpoints (Cey, 2001; Karagiorgi & Symeou, 2005; Perkins, 1999). Together with other learners, an individual learns through interaction, shared responsibility, cooperation, and collaboration with other learners (Cey, 2001; Loyens et al., 2006; Karagiorgi & Symeou, 2005; Powers, 2004), leading to a shared reality (Karagiorgi & Symeou, 2005). A study by McDougall and Boyle (2004) found a rich source of programming knowledge and skill in peers (learner-as-teacher) and wider society and found that learners are not confined to teacher-provided knowledge. Research also shows that learning occurs when learners have to provide explanations and justify understandings to peers in small group situations – new knowledge is promoted by encouraging learners to verbalise (discuss, explain, and evaluate) their ideas and procedures (Hendry, 1996).

2.2.3 Prior knowledge

Learners are always constructing understanding and knowledge for their experiences, and this is influenced by the cognitive lens of what is already known and what currently makes sense (Confrey, 1990; Powers, 2004; Von Glasersfeld, 1992). New knowledge is, thus, constructed on the base of prior knowledge, experiences, goals, and beliefs, and these form the filters through which learning occurs (Harris & Graham, 1994; Karagiorgi & Symeou, 2005). Teachers can get learners to make their current understandings verbal (through talking about their understandings) and visible (through written artefacts) and need also to encourage reflection (Ben-Ari, 1998; Clement, 2004).

2.2.4 Authenticity

If knowledge is constructed in the mind of the learner (as noted by Bodner above), this implies that knowledge must be meaningful, understood, and believed (and not just learnt by rote without understanding) (Bodner, 1986; Confrey, 1990; Von Glasersfeld, 1995). Such ownership of knowledge needs problems that are authentic and, thus, often ill-structured because of the complexities associated with real-life, everyday problems for the learning to have relevance and meaning (Karagiorgi & Symeou, 2005; Loyens et al., 2006). Note, though, that authenticity and contextualised meaning do not necessarily exclude explicit instruction or structure (Harris & Graham, 1994).

The emphasis on authentic problem-solving activities means less “fragile” knowledge (Perkins & Martin in Robins et al., 2003, p. 151) – knowledge that has been forgotten, learnt but not used, or learnt but used inappropriately (Perkins, 1999; Robins et al., 2003). Because of the disorder that goes along with the complexities of authentic environments and problems, learners and teachers will also need to learn that there are lessons in failure as well (Cey, 2001).

2.2.5 Teacher's role

"Under the old education the teachers taught subject matter; today they are expected to teach children" (Miller, Courtis & Watters in Harris & Alexander, 1998, p. 115). Although this was originally written in 1931, the sentiment is still valid and the role of the teacher still under debate.

Learning and teaching are not the same thing, and even when teaching may be happening, learning may not occur at the same time (Karagiorgi & Symeou, 2005). A paradigm shift is needed in the role of the teacher (Cey, 2001). To provide learning experiences and an environment that will lead to learner reflection and knowledge construction, teachers need to have some model of the current levels of understanding of the learners, and this is gained through learner talk rather than teacher talk (Bodner, 1986; Confrey, 1990; Von Glasersfeld, 1995). Teachers working from a constructivist viewpoint, then, guide learners to an understanding of what is currently regarded by society as viable knowledge (Tobin & Tippins, 1993).

There are some constructivist views that see teaching (as direct, explicit instruction) as a dirty word (Harris & Graham, 1994). Such approaches leave control over the construction of content, what is to be studied when and how, to the learner (Karagiorgi & Symeou, 2005). The choice between instruction and knowledge construction is a false one, and teaching lies in a balance between the two extremes (Harris & Alexander, 1998). However, in ill-defined problem environments such as computer programming, where complex issues that cannot always be neatly described in concise or complete ways lead to competing approaches (Reed, 2002), the teacher needs to provide appropriate learner guidance. In such environments, allowing learners to structure their own learning is "not a great virtue but abdication of our responsibility as teachers" (Merrill in Karagiorgi & Symeou, 2005, p. 22). Teachers, then, do have an active role to play in guiding learners' knowledge construction (Hendry, 1996).

2.3 Critics of constructivism

It does need to be pointed out, though, that there are critics of constructivism, especially when its theory of knowledge (or epistemology) is carried to the extreme (Andrew, 2004; Ben-Ari, 1998). Constructivism is, in a sense, a theory about the limits of human knowledge, believing that all knowledge is essentially a product of our own cognitive experiences (Confrey, 1990). Thus, it denies the possibility of certain knowledge and focuses on what we experience and what knowledge is proven to be viable about our environment (Von Glasersfeld, 1992).

However, without denying the existence of a reality, our knowledge is constructed so as to fit our experience of reality in a personal and subjective way (Bodner, 1986; Tobin & Tippins, 1993). Thus, learners are not free to simply construct any knowledge (Karagiorgi & Symeou, 2005), but only knowledge that is viable and useful to survive within the reality that is experienced every day. Put another way, learners are "not seeking truth in constructivism, but rather constructing models that could correctly explain the reality" (Lui et al., 2004, p. 73).

2.4 Teaching programming constructively

Constructivism advocates that for successful construction of new concepts, the teacher must understand the learner's level of understanding and existing cognitive structures (Clement, 2004; Feldman & Minstrell, 2000). For computer programming education, this can be problematic, as novice programmers often do not have an effective mental model of a computer on which to construct new material, leading to haphazard constructions (Ben-Ari, 1998). Lui et al. (2004) go as far as suggesting that for weak learners learning programming, all prior mental models should be considered non-viable and that such models be constructed from scratch, assuming nothing. Further, incorrect constructions usually lead to immediate, and discouraging, feedback from a compiler (Ben-Ari, 1998). Thus, these together lead learners to believe computer programming to be difficult and frustrating (Ben-Ari, 1998; Lui et al., 2004).

The gap between the theory of constructivism and educational practice is not easy to bridge (Gijbels et al., 2006). Below is a set of techniques that would be included in a constructivist approach to the learning of computer programming.

2.4.1 Mental model of a computer

Some understanding of the low-level organisation and operation of a computer is considered important for those wanting to learn computer programming (Powers, 2004). Thus, learners need to actively construct a mental model of a computer to avoid haphazard constructions (Ben-Ari, 1998). Robins et al. (2003) note that there are many studies that have identified the central role played by a meaningful model of the computer, as it provides the learner with an understanding of the behaviour of running programs.

2.4.2 Apprenticeship approach

An apprenticeship approach can be used where learners are provided with scaffolding such as completed or partially written programs, program comments, and teacher demonstrations (Clement, 2004; Kölling & Barnes, 2004; Van Gorp & Grissom, 2001; Wulf, 2005). Such scaffolding operates in Vygotsky's zone of proximal development, moving learners from what they can do without help to where they could be with help (Ben-Ari et al., 2004).

2.4.3 Problem-driven approach

Learning should be built around a problem-driven approach that presents practical and realistic programming problems to which real solutions can be found (Kölling & Barnes, 2004). New programming constructs would, thus, be introduced within a context of where and why they are used.

2.4.4 Working in class

Group assignments and in-class exercises can be used to prevent premature attempts at individual program creation, which could lead to frustration and a reduction in self-confidence (Ben-Ari, 1998).

2.4.5 Socratic questioning

The Socratic method of questioning uses questions to guide the learners to their own answers rather than simply giving learners answers to their questions (Clement, 2004). Further, when using this method, the temptation to tell the learner what has to be done to solve a problem must be avoided, as evidence indicates that correcting learners and giving them answers is not a productive strategy (Tytler, 2002). Such questioning allows the teacher to judge a learner's level of understanding and thinking and what basis the learner will be using to construct knowledge (Clement, 2004). Also, telling learners they are wrong may discourage them from honestly verbalising their thinking in the future (Tytler, 2002).

2.4.6 Paper first

Learners do their problem solving on paper first so that negative feedback from a compiler, which can lead to impatience and lessening of confidence, is minimised (Clement, 2004; Lui et al., 2004). This then creates a low-risk environment for weaker learners in which to learn.

2.4.7 Manipulatives and role playing

The use of physical props, or manipulatives, provides learners with a physical model that can help them understand the process that a program would follow (Powers, 2004). Such props augment the learning experience and allow for the construction of viable mental models (Astrachan, 1998). For example, paper squares can be used for demonstrations of arrays in sorting algorithms (Clement, 2004), and flash cards representing the parts of a computer can also be used to teach computer architecture (Powers, 2004).

Role playing can be seen as a special form of the use of manipulatives. Learners can role-play the responsibilities of parameter values in the use of functions and procedures in modular programming (Gonzalez, 2004) and, in so doing, act out the transfer of values between calling and called functions.

2.4.8 Multiple examples

Analogies or metaphors are generally popular instructional tools, but can lead to non-viable mental models, as programming is human-built and often does not have parallels in the real world (Lui et al., 2004). Multiple examples of the programming structure to be learnt will, thus, be a better route to provide learners with usable mental models of structure and program behaviour.

2.4.9 Learning cycles

Several learning cycles can be used in the constructive classroom (Tytler, 2002), and Sunal (2007) reviews some of these. These cycles are designed to assist learners in experiencing new ideas in a safe, positive learning environment, coming to terms with what they already know, constructing new knowledge on this, and putting new learning to work in new and novel situations (Sunal, 2007). This is achieved by involving learners in a sequence of activities that exposes them to a new idea or skill, through guided presentation and explanation, to expansion of the idea or skill through practice (Sunal, 2007). Such a sequence would represent one lesson, although it may take several instructional periods to complete the whole lesson (Sunal, 2007). One such cycle, the Karplus learning cycle, will be covered in greater depth in section 2.5.

Often constructivist teaching is anomaly driven – confronting learners with situations that make inconsistencies in the learners' current knowledge visible and, thus, encourages them to find alternative knowledge structures (Perkins, 1991). A three-phase learning cycle that uses this approach is McDermott's predict-confront-resolve learning cycle, which can effectively handle learner misconceptions (Clement, 2004; McDermott, 1991). Learners, firstly, predict what they expect to happen when their

programs run (using their knowledge and understanding of programming). They are then confronted with the results of their program and compare these with their expectations. Finally, they need to find the source of the differences and resolve them, in such a way bringing their understanding in line with how the program actually behaves and runs.

2.5 The Karplus learning cycle

The Karplus learning cycle is influenced by Piagetian theories of development (Sunal, 2007) and be can be used when introducing new concepts (Clement, 2004) and nurturing curiosity (Cey, 2001). It is a three-phase learning cycle made up of exploration, concept introduction, and concept application (Karplus, 1980). How this learning cycle can be used in learning computer programming will be explored below.

- In the exploration phase, a learner learns through his or her autonomous action and reaction, testing ideas and reacting to the feedback gained (Karplus, 1980; Sunal, 2007). During this time, there is minimal guidance from the teacher. A topic in computer programming is, thus, introduced through experimentation with an actual running program. The learners should be presented with the concept cold with no preliminary explanation, forcing them to come to terms with what the program is doing (Clement, 2004), thus also preventing learners from constructing inappropriate mental models of the concepts from previous knowledge (Lui et al., 2004).
- Concept introduction should always follow exploration and should relate directly to what was experienced there (Karplus, 1980). This phase leads learners to new reasoning patterns that can be used to explain the new concept, and it may be introduced via any medium, including the teacher (Karplus, 1980). In this phase, the teacher is more active (Sunal, 2007). In terms of learning computer programming, concept introduction provides the formal terminology for the programming structures that have been used. At this point, it is important to note that the Karplus learning cycle does not exclude direct instruction, and it usually occurs in this second phase of the cycle (Clement, 2004). There are, however,

some constructivist approaches that specifically reject direct or explicit instruction, considering it not to be desirable, even harmful (Harris & Graham, 1996b).

- In the concept application phase, the concept is applied to new situations, thus extending the range of applicability of the new concept or skill (Sunal, 2007). Such practice also allows the new knowledge to stabilise (Karplus, 1980). In this last phase, the new concept is applied in practice programs.

Research has shown that this approach can improve content comprehension and encourage learning and thinking (Clement, 2004). It also meets the requirements of constructivist teaching in that learning must be experiential to be effective (Powers, 2004). Glasson (in Tytler, 2002) uses social constructivist insights to suggest a variation to the Karplus cycle (see Figure 1) that spirals and is interactive. It places more emphasis on class discussion and negotiation of meaning and understanding.

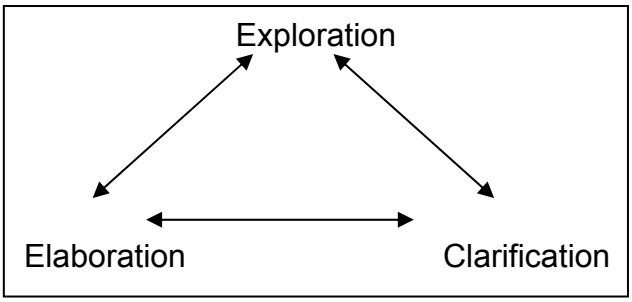

Figure 1 Glasson's learning cycle (after Tytler, 2002, p. 31)

2.6 Conclusion

Most of the techniques in sections 2.4 and 2.5 above are aimed at getting the learner to grapple with the concepts before learning the formal terms for them (Clement, 2004), and as such, constructivism plays a role in modelling the processes that learners follow in creating an understanding of computer programming concepts (Pears et al., 2002). Also, evidence seems to suggest that the rote learning of program and syntax rules, in contrast to constructing understandings, may contribute to learner confusion (Fleury, 1991).

Wadsworth (1996) and Jonassen (1991) note that constructivism is not a theory of education and that it further does not hold all the answers to our instructional problems. Confrey (1990, p. 20) concludes that "we have learned to think of constructivism as a referent, not the only referent …, but nonetheless an important one". Thus, constructivism can provide another set of concepts that can be used to guide our debates about, and understandings of, computer programming education (Ben-Ari, 1998).

The courses on which this study is based used constructivist approaches to get across the concepts of introductory computer programming to learners with ADHD. In particular, the Karplus learning cycle was used to structure the concept presentation. It has been noted, however, that the challenges faced by learners with special educational needs are complex and that no one intervention can address their learning needs (Harris & Graham, 1996b). The question is, then: is constructivism (as expressed in the Karplus learning cycle) one intervention that can address the needs of learners with ADHD? I will seek an understanding of ADHD next before taking this question further.

Chapter 3 Understanding ADHD

3.1 Introduction

Before I can begin to address the problem of teaching programming constructively to children with ADHD, I need an understanding of constructivism (Chapter 2) *and* a thorough understanding of what ADHD entails. In this chapter, I survey the literature about what is known about ADHD and what is believed to help these children learn. Raggi and Chronis (2006) note that there is a strong link between ADHD and academic impairment and that the academic difficulties experienced by children with ADHD are significant. It is, therefore, important that any approach that may help children with ADHD learn computer programming be utilised to enhance learning in the classroom.

I will present a general background to ADHD first, examining the diagnostic criteria that are used to define it and its general characteristics. This background will include arguments about its existence, how it is diagnosed, and its causes, incidence, and link to learning disabilities. Several learning strategies have been shown to help learners with ADHD learn, and I have divided these into five intervention categories: instructional, behavioural, structural, organisational, and medical. These categories will each be discussed in turn. Finally, I will address the question as to whether constructivism can be used as a teaching methodology alternative.

3.2 What is ADHD?

The latest edition of the *Diagnostic and Statistical Manual of Mental Disorders* (DSM-IV) (American Psychiatric Association, 1994, pp. 83-85) lists the diagnostic criteria of ADHD:

A. Either (1) or (2):

(1) six (or more) of the following symptoms of **inattention** have persisted for at least six months to a degree that is maladaptive and inconsistent with developmental level:

Inattention

(a) often fails to give close attention to details or makes careless mistakes in schoolwork, work, or other activities

(b) often has difficulty sustaining attention in tasks or play activities

(c) often does not seem to listen when spoken to directly

(d) often does not follow through on instructions and fails to finish schoolwork, chores, or duties in the workplace (not due to oppositional behavior or failure to understand instructions)

(e) often has difficulties organizing tasks and activities

(f) often avoids, dislikes, or is reluctant to engage in tasks that require sustained mental effort (such as schoolwork or homework)

(g) often loses things necessary for tasks or activities (for example, toys, school assignments, pencils, books, or tools)

(h) is often easily distracted by extraneous stimuli

(i) is often forgetful in daily activities

(2) six (or more) of the following symptoms of **hyperactivity-impulsivity** have persisted for at least six months to a degree that is maladaptive and inconsistent with developmental level:

Hyperactivity

(a) often fidgets with hands or feet or squirms in seat

(b) often leaves seat in classroom or in other situations in which remaining seated is expected

(c) often runs about or climbs excessively in situations where it is inappropriate (in adolescents or adults, may be limited to subjective feelings of restlessness)

(d) often has difficulty playing or engaging in leisure activities quietly

(e) is often “on the go” or often acts as if “driven by a motor”

(f) often talks excessively

Impulsivity

(a) often blurts out answers before questions have been completed

(b) often has difficulty awaiting turn

(c) often interrupts or intrudes on others (for example, butts into conversations or games)

B. Some hyperactive-impulsive or inattentive symptoms that caused impairment were present before age 7 years.

C. Some impairment from the symptoms is present in two or more settings (for example, at school [or work] and at home).

D. There must be clear evidence of clinically significant impairment in social, academic, or occupational functioning.

E. The symptoms do not occur exclusively during the course of a Pervasive Developmental Disorder, Schizophrenia, or other Psychotic Disorder, and are not better accounted for by another mental disorder (for example, Mood Disorder, Anxiety Disorder, Dissociative Disorder, or a Personality Disorder).

Three subtypes are recognised (American Psychiatric Association, 1994):

1. Attention-deficit/hyperactivity disorder, combined type: if both criteria A (1) and A (2) have been met for the past six months.
2. Attention-deficit/hyperactivity disorder, predominantly inattentive type: if criterion A (1) has been met, but criterion A (2) has not been met for the past six months.
3. Attention-deficit/hyperactivity disorder, predominantly hyperactive-impulsive type: if criterion A (2) has been met, but criterion A (1) has not been met for the past six months.

Naparstek (2002) further discusses the three main characteristics of the ADHD symptoms by noting the following:

- Inattention: although this involves difficulties with keeping the learner's attention focused and being able to shift the focus of attention as necessary, ADHD could be seen as a "boredom disorder", as such learners are often able to focus their attention on tasks or activities that are interesting to them (such as video games). This characteristic will be returned to later.
- Hyperactivity: he likens this characteristic of ADHD to the bunny in the TV commercial that keeps on going and going.
- Impulsivity: the learner has difficulties with being able to think before acting and could be described as a car with faulty brakes.

Although the DSM-IV description given above identifies three forms of the disorder:

- the combined type,
- the predominantly inattentive type, and
- the predominantly hyperactive-impulsive type,

both Naparstek (2002) and Bester (2000) note that the core problem with ADHD children is the inability to pay attention. Thus, they both focus on only two forms of ADHD:

- The combined type (the whole syndrome of attentiveness, hyperactivity, and impulsivity) – the typical ADHD child, which Bester (2000) characterises as tornadoes; and
- the predominantly inattentive type – which is often called ADD (ADHD without the hyperactivity/impulsivity), which Bester (2000) characterises as dreamers.

It needs to be noted, however, that there is some controversy about whether ADHD is a disorder of control of attention or one of sustained attention, and the root cause of these attention difficulties remains unclear (Wilding, 2005). There are also indications that impaired visual-orientating (Savitz & Jansen, 2005) and manipulation of spatial working memory (Castellanos, Sonuga-Barke, Milham & Tannock, 2006) may be associated with ADHD.

Bester (2000) takes the categorisation one step further by seeing a continuum from ADD to ADHD. She places a third group of children in between these two extremes (characterised as fidgeters, as they show all the signs of inattentiveness, but only the first symptom of hyperactivity, and none of those under impulsivity). This can be shown diagrammatically as in Figure 2.

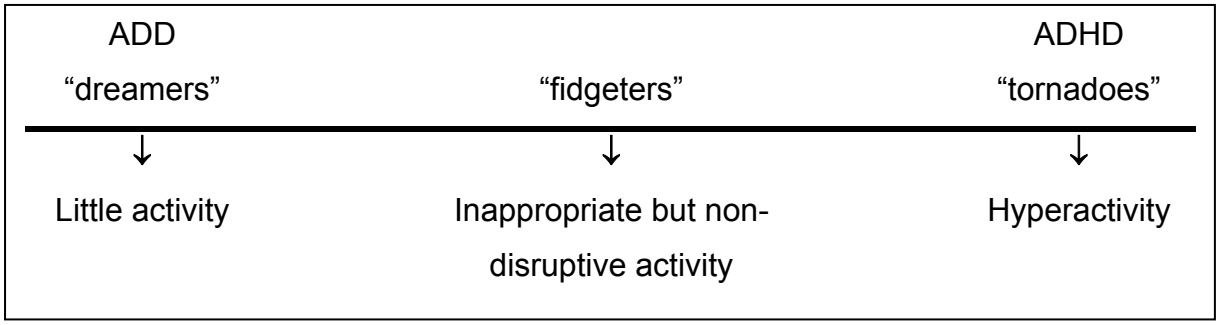

Figure 2 An ADD-ADHD continuum (after Bester, 2000, p. 34)

As understandings and definitions change around this disorder, it seems increasingly unlikely that it represents a qualitatively distinct subgroup (Wilding, 2005), and as research progresses, further refinements are to be expected. ADHD is then, maybe, an umbrella construct that has multiple causes with overlapping cognitive profiles (Castellanos et al., 2006).

There are authors who question, or argue against, the existence of ADHD (Armstrong, 1996; Smelter, Rasch, Fleming, Nazos & Baranowski, 1996) and say that it is a "highly debatable and pseudomedical concept" (Kohn in McEwan, 1998, p. 4). However, the behaviours and learning difficulties (with their implications) exist, and though ADHD has been known by other names in the past, the symptoms do remain constant and occur all over the world in similar patterns ("Attention Deficit Disorder", 2006; Bester, 2000; McEwan, 1998). Also, the concepts around ADHD are constantly changing as experience and research cast more light on what it is, what causes it, and how best to help learners who have these problems (McEwan, 1998).

3.3 Diagnosing ADHD

Bester (2000) dislikes the term "diagnosis", as it implies that the focus is on what is defective or disabled in the child, rather than on the particular problems and difficulties a child has or the child's strengths and potential (Armstrong, 1996). Restak (in Gigout-Hues, 2006) goes further and sees it not so much as a disorder as a different cognitive style. Also, not all the symptoms apply to each child, and not only will the symptoms appear in different combinations, but they will also vary in

degree (Bester, 2000; McEwan, 1998; Rief, 1993). Each child, therefore, will be unique. Also, as most of these behaviours are typical (and normal) at various developmental stages in all children, it is the number of these behaviours over an extended period of time when developmentally inappropriate that leads to a diagnosis of ADHD. It is important to note, also, in accordance with the DSM-IV description, that these symptoms must be present in at least two different situations (Naparstek, 2002) – in other words, such symptoms cannot only be prevalent at school and must be seen in other situations as well (at home, for example).

Keeping this in mind, such children must be identified and supported if they are to reach their potential. Diagnosis of ADHD requires a multidisciplinary, wide-ranging approach (McEwan, 1998; Ross & Ross, 2006; Shore, 1998; Smelter et al., 1996), although the actual diagnosis is made by a medical doctor, psychologist, or psychiatrist (McEwan, 1998; Naparstek, 2002). Schoolteachers have a role to play in that they can provide valuable information to the medical professionals who make the final diagnosis, as it is in the school situation where the most demands will be made on a child's levels of attention (Naparstek, 2002). It is, however, not the teachers' job to make such a diagnosis (Naparstek, 2002).

Several authors (Castellanos et al., 2006; Naparstek, 2002; Sayal, Goodman & Ford, 2006; Shore, 1998) point out that diagnosing ADHD is not a simple task, as there is no single reliable test that can be used to diagnose it. This makes diagnosis controversial. Further, as a diagnosis is based on multiple measures, it is subjective and is only as good as the person who made the diagnosis – an art more than a science (Armstrong, 1996; Naparstek, 2002; Smelter et al., 1996).

It has also been noted (Bester, 2000; Naparstek, 2002) that children with the predominantly inattentive type of ADHD (or ADD) are often better behaved than those with the combined type and are, thus, less likely to be identified and diagnosed. Also, those with more advanced intellectual abilities are often able to compensate for the problem, and it remains undiagnosed.

As the diagnosis of ADHD has become more common ("Working", 2006), concerns about misdiagnosis have been raised, particularly in the United States, although, in

other parts of the world, authors believe that most children with ADHD still remain undiagnosed and untreated (Sayal et al., 2006). One study (in the United Kingdom) found that one of the main barriers to the identification of ADHD was that although parents recognised that there was a problem, it was often seen as a behavioural or learning difficulty, and few saw it in terms of hyperactivity (Sayal et al., 2006). Some authors further argue that a diagnosis of ADHD relieves the parents of having to deal with discipline problems in their children, and the fact that some dysfunction has been found means that they can take on the role of victim rather than deal with social censure for badly behaved children (Smelter et al., 1996). This has been countered by parents who argue that they would rather the fault lay with their parenting skills than have to admit that there was something wrong with their child (Thompson, 2006).

3.4 Causes of ADHD

The causes of ADHD are not fully understood (Rief, 1997). Though it was originally seen as an environmental problem (poor parenting, for example) (McEwan, 1998), recent scientific research points to biological and physiological differences in the neurochemistry of the brain (in terms of a lack of neurotransmitter production in the frontal lobe of the brain) (Bester, 2000; McEwan, 1998; Naparstek, 2002). Currently, authors point to differences in higher order, or executive, functioning, which has to do with inhibitory control, attentional regulation, self-regulation, goal-directed action, and working memory – as the root of the disorder ("Attention Deficit Disorder", 2006; Castellanos et al., 2006; Raggi & Chronis, 2006; Savitz & Jansen, 2005; Wilding, 2005). There appear to be cognitive and affective aspects to executive function, where the former is related to attentional problems and the latter to the hyperactivity, and delayed reward aversion, of ADHD (Castellanos et al., 2006).

Although the neurobiological basis for the disorder is fairly widely accepted, the reason for its occurring remains unknown. It does, however, appear to be genetic in nature in that it is primarily inherited from one or both of the child's parents ("Attention Deficit Disorder", 2006; McEwan, 1998; Naparstek, 2002). Further, it is

accepted that the environment may exacerbate rather than cause ADHD (McEwan, 1998).

Taking this last point a step further, Weaver (1994) wants to move the locus of the problem away from the child alone and argues for a more systems-theory approach that sees causation as multidimensional. The implication is that the rigid structures of many school systems (Weaver, 1994) are part of the problem rather than just bringing it into sharper focus or making it worse.

An understanding of the causes of ADHD is important for examining the interventions that are attempted to assist children with ADHD in schools to learn more effectively.

3.5 The incidence of ADHD

It has been estimated that between 3% and 5% of the population has ADHD (American Psychiatric Association, 1994; DuPaul & White, 2006; Rief, 1997, Westwood, 2003), although percentages as high as 8% ("Attention Deficit Disorder", 2006) and 20% (Armstrong, 1996) have been quoted.

It is also believed that more boys than girls have ADHD ("Attention Deficit Disorder", 2006), although it is generally believed that as boys tend to be of the predominantly hyperactive and combined types and girls of the predominantly inattentive type, ADHD among girls is often overlooked and not identified. The ratios (of boys to girls) are given as between 3:1 and 9:1 (American Psychiatric Association, 1994; Bester, 2000; McEwan, 1998; Rief, 1997; Thompson, 2006).

Although ADHD is often referred to as a childhood or school-age disorder, there is evidence that it is a lifelong disorder, is present from birth, and continues into adulthood (Bester, 2000; Rief, 1997; Wolraich, 2006). The DSM-IV definition included above (in section 3.2) notes that the symptoms of ADHD should be present before the age of seven, and with the increase in pre-school learning, calls have been made for the earlier identification of ADHD (Wolraich, 2006). It has been

noted, however, that the symptoms of the disorder may change from childhood, through adolescence, to adulthood (Rief, 1997).

3.6 ADHD and learning disabilities

It should be noted that ADHD is not a learning disability. A learning disability may be defined as

> a generic term that refers to a heterogeneous group of disorders manifested by significant difficulties in the acquisition and use of listening, speaking, reading, writing, reasoning or mathematical abilities, or of social skills. These disorders are intrinsic to the individual and presumed to be due to central nervous system dysfunction. Even though a learning disability may occur concomitantly with other handicapping conditions (e.g., sensory impairment, mental retardation, social and emotional disturbance), with socio-environmental influences (e.g., cultural differences, insufficient or inappropriate instruction, psychogenic factors), and especially with attention deficit disorder, all of which may cause learning problems, a learning disability is not the direct result of those conditions or influences. (National Joint Committee for Learning Disabilities definition in Kavale & Forness, 1992, pp. 13-14)

Many children with a learning disability simulate ADHD, as they develop avoidance behaviours similar to ADHD (and often have attention problems even if they do not meet the diagnostic criteria for ADHD). Also, children with ADHD often struggle to maintain the required attention levels to learn effectively and, thus, appear to have a learning disability (Bester, 2000; Mayes, Calhoun & Crowell, 2000). McEwan (1998, p. 16) summarises the difference between ADHD and learning disabilities as follows:

> In layman's language, a learning disability means there is a major discrepancy between a child's ability and performance. Although children with ADHD (without accompanying learning disabilities) often have a large discrepancy between ability and performance, the reasons for this discrepancy are different. Children who are distractible and inattentive will have a hard time learning because of their inattention. But if a child also has a learning disability in association with

ADHD, merely getting them on task will not be sufficient. They will still have a problem with the learning task.

Further, ADHD symptoms should also occur in a situation outside the school environment; this is not true of a learning disability, which is usually seen as an underachievement in the school setting.

However, there is a high co-morbidity for learning disabilities and ADHD (Bester, 2000; DuPaul & White, 2006; Mayes et al., 2000; McEwan, 1998; Raggi & Chronis, 2006), and between 25% and 50% of children with ADHD have a learning disability as well. Some research has shown that up to 70% of children with ADHD have a learning disability as well – particularly in written expression (Mayes et al., 2000) − whereas others have found the overlap as high as 92% (Savitz & Jansen, 2005). It is possible, therefore, that even though learning disabilities and ADHD are not the same thing, they are interrelated, and attention and learning problems may be on a continuum, usually occurring together (Mayes et al., 2000). However, some argue that ADHD may be a cognitive defect that manifests as learning problems rather than inattention and impulsivity (Savitz & Jansen, 2005).

3.7 Learning strategies for learners with ADHD

Parents and teachers should not lower their expectations of children just because they have been diagnosed with ADHD, as they will then simply meet the lowered expectations (Smelter et al., 1996). All children must be helped to reach their potential – their cognitive styles should be embraced rather than fought against and their strengths built on, while recognising their limitations (Gigout-Hues, 2006; Ross & Ross, 2006) – thus the learning strategies below. Tsai and Tsai (2003) found that the truth that better learning strategies lead to better learner outcomes in various other school subjects is true also for learning about computers. Thus, the various general strategies that are used in teaching and learning, in general, will also apply to the teaching and learning of computer programming.

Much of the literature on teaching children with ADHD simply presents an array of useful methods for getting, focusing, and maintaining children's attention and managing behaviours (see Bester, 2000; Hallowell and Ratey, 1999; McEwan, 1998; Rief, 1993). Although there is a fair amount of agreement on what techniques will help ADHD children learn, various authors categorise these strategies in different ways (if at all).

Many of the interventions are based on modifying the ADHD child's behaviour almost as though the locus of the problem lies solely within the child. The more systems-theory approach taken by Weaver (1994) sees causation as multidimensional, which leads to interventions based on transactional, constructivist, and holistic paradigms of learning. This means that managing ADHD is about meeting the needs of individual learners within the school system (Armstrong, 1996). The systems approach recognises that the issues are broader than just those of a child with problems and tries to address the very structures of the classroom that exacerbate the ADHD (Weaver, 1994).

Approaches to managing ADHD children have generally been multifaceted (Brand et al., 2002; Rief, 1993; Smelter et al., 1996; Weaver, 1994):

- Cognitive therapy approaches (helping to develop learner self-control)
- Behaviour modification approaches (using external control to manage the learner's behaviour)
- Medical approaches (based on biochemical effects)
- Counselling (for both family and individual to learn coping techniques)
- Parent education (so that they can help the child and be an effective advocate)
- Social skills training (an area where ADHD children have particular problems)
- School interventions (environmental, instructional, and behavioural)
- Physical outlet (participation in non-competitive sports to help the learner focus and concentrate)

The following intervention categories (based on areas that can be addressed to maximise learning by ADHD children) will be used in this study:

- Instructional
- Behavioural
- Structural and environmental
- Organisational
- Medical

Most, if not all, of the interventions and ideas proposed below would apply to any classroom (DuPaul & White, 2006), but are highlighted by authors as of particular importance when teaching a learner with ADHD.

3.7.1 Instructional focus

Rief (1993) notes that ADHD children need creative, engaging, and interactive teaching strategies. Such strategies include the use of a variety of approaches – multisensory teaching strategies, cooperative learning, recognition of learning styles, and the theory of multiple intelligences – all within a structured flow of these activities.

Beginning with the flow of instruction, there is reference to the logical structure of instruction that will support learning by children with ADHD. This flow begins with a form of lesson cueing and setting the scene for the lesson, as well as reviewing previous learning (McEwan, 1998) – this can include letting learners know what is meant to be learnt from the lesson (McEwan, 1998). The structure of the lesson itself should be kept logical and sequenced, ensuring that the lesson does not drift off on tangents that can only serve to lose or confuse a learner (McEwan, 1998). More important information should be presented when ADHD learners are at peak performance (earlier in the morning rather than later, or two hours after medication has been taken) (McEwan, 1998; “Working”, 2006), key information should be summarised (McEwan, 1998), and periodic breaks should be scheduled (McEwan, 1998). As transitions, and non-instructional time, can be very problematic for

children with ADHD (Rief, 1993), learners should be given advance warning that a lesson is about to end, thus allowing time for the transition and time to get organised (Burgess, 2003; McEwan, 1998).

Various strategies should be employed to assist ADHD learners in learning, and varying lesson presentation strategies (that allow for alternating periods of sitting still and more participatory involvement) have been found to be useful (McEwan, 1998). Particularly, using multisensory instruction that targets all senses and being aware of the theory of learning styles and intelligences that highlights different learners' preferences when learning will enable teachers to be better prepared to help all kinds of learners (McEwan, 1998; Naparstek, 2002; Rief, 1993). These strategies are also useful when material has to be retaught, as they allow the teacher to use a different mode and sense in the process of helping the learner (Rief, 1993).

Several techniques, many of which implement the strategies mentioned above, can be used during instruction. Some of the more common techniques are noted below:

- The use of cooperative learning (as opposed to competitively or individually structured learning situations) promotes several learning outcomes and is shown by research to be more beneficial than the two alternatives (although all three have a place in the classroom) (Rief, 1993). Cooperative learning allows learners to verbalise their thoughts (and their understanding), as well as learn to work in a group (thereby building social interaction skills) (Rief, 1993). It also leads to less disruptive and more on-task behaviour (Rief, 1993).
- Chunking the material to be learnt into smaller, more manageable chunks assists ADHD learners by fitting learning into sessions that a child with attention and memory problems can handle (Burgess, 2003; Naparstek, 2002; Rief, 1993).
- Mnemonic devices can be used to aid memory (McEwan, 1998). Examples include those that are used to help spelling (rhythm helps your two hips move = rhythm) and the order of items in a list (such as the colours of the rainbow: Richard of York gave battle in vain = red, orange, yellow, green, blue, indigo, violet).

- Cloze notes (which provide an outline of the content to be learnt with words missing) and full-text notes (which allow for highlighting or underlining) can help ADHD learners focus attention during direct instruction (McEwan, 1998; Rief, 1993).
- A teacher can provide a completed example that could serve as an example of what is expected of the learner (McEwan, 1998). Also, a folder of the learner's best, completed work can be used as a comparative standard to motivate the learner and against which future performance can be evaluated (McEwan, 1998).
- For learners who may struggle with a particular form of information presentation, teachers can give ADHD learners a variety of options for reporting information and completing assignments (for example, audio/videotapes, displays, and presentations) (DuPaul & White, 2006; McEwan, 1998; Westwood, 2003).
- Working within the child's field of interest can be used to motivate learning (Bester, 2000).
- Vary the pace and type of activity in lessons ("Working", 2006).

In the field of questioning, teachers should teach learners information-locating strategies that learners can use by themselves when unsupervised (McEwan, 1998) – this allows them to complete tasks without supervision. Further, questioning allows learners to become active participants in the learning process (Naparstek, 2002). When questions are being used, teachers should ensure that there is ample wait time to allow the ADHD learner time to organise a verbal response (McEwan, 1998; Rief, 1993).

A feature that must be remembered in all instruction of ADHD learners is that all teaching techniques must be constantly revised – what worked in the classroom last month is unlikely to work next month (McEwan, 1998) – thus the emphasis on a variety of strategies and techniques.

3.7.2 Behavioural focus

Many of the techniques proposed to assist children with ADHD in learning are based on behaviour modification – both direct, where the teacher controls the child using

various strategies, and indirect, using strategies to get the child to control or moderate his or her own behaviour. A diagnosis of ADHD should not become a licence to allow the child to do as he or she pleases (Smelter et al., 1996). Included in this section will be techniques for giving instructions and other techniques for helping such learners to focus.

Rief (1993) notes that behavioural difficulties in a class are often exacerbated when learners are undirected, such as during transition times. Thus, a teacher should anticipate problems in particular situations and activities and plan ahead to avoid them (McEwan, 1998; Rief, 1993). Similarly, a teacher should watch for signs that indicate that a learner's thoughts are straying and then bring him or her back to the present and the task at hand (Bester, 2000).

One way in which a teacher can help bring a learner's attention back to the work that has to be completed is to agree on a set of silent preventative cues with the child (Gigout-Hues, 2006; Rief, 1993) – these can then be used to help a learner return to the task without embarrassing the child in front of the rest of the class. Related to this is the use of proximity control (McEwan, 1998; Rief, 1993) – moving closer to the child when he or she is beginning to drift. Moving around the class during a lesson will also help a teacher to pick up cues that something has not been understood and to correct it before it leads to a behaviour problem (McEwan, 1998). Also, a teacher can seat children with ADHD near appropriate role models in the class who can model "on-task" behaviour for them (Bester, 2000; Rief, 1993), ensuring further that distractions around them are limited, without isolating them (Gigout-Hues, 2006; "Working", 2006).

A point that is often made in the management of ADHD learners in a classroom is that there be clear rules and expectations that define the learner's space and limits (Burgess, 2003; Rief, 1993). The rules need not only be taught and reviewed regularly (McEwan, 1998), but the rationale behind them must also be clear (Rief, 1993). It is, however, noted that there should only be a few, simple, positively stated rules (Westwood, 2003) that are displayed in a highly visible area. Not only must the rules be clear, but there must be clear consequences that are implemented consistently within an organised system (McEwan, 1998; Rief, 1993).

How instructions are given is also important in a classroom that caters for learners with ADHD. A teacher should make sure that he or she gets the attention of the class before giving instructions, such as through making eye contact and waiting until the class is quiet (Bester, 2000; Burgess, 2003; McEwan, 1998; Rief, 1993). Frequent eye contact can also be used to maintain attention (McEwan, 1998). When instructions are given, they should be brief, specific, direct, logical, and sequential (Bester, 2000; Gigout-Hues, 2006; McEwan, 1998), given one at a time where possible (Bester, 2000; Burgess, 2003), making sure not to overwhelm the learner (Rief, 1993).

Bester (2000) notes that there are two basic principles of behaviour modification: reinforcing good behaviour and imposing negative consequences for bad behaviour. Once instructions have been followed, concrete reinforcement needs to be given for work completed (McEwan, 1998). Authors note that there is no substitute for positive reinforcement and that it is the best behaviour management strategy available, as it also builds self-esteem and self-respect (Gigout-Hues, 2006; Rief, 1993; “Working”, 2006). To have this effect, though, the praise that is given must be specific and legitimate (McEwan, 1998; Rief, 1993). In a sense, then, the teacher wants to catch the learner doing what is required of him or her (Bester, 2000; Rief, 1993) and praise him or her for his or her behaviour. Note that this praise should not just be for being on task and attending, but also for appropriate social behaviour (McEwan, 1998). At the other end of the spectrum, reprimands should be brief and directed at the unwanted behaviour rather than the learner (McEwan, 1998). Also, rewards should outnumber punishments by two or three to one (Bester, 2000).

Cognitive behaviour modifications try to address behaviour difficulties by putting the child in control of the behaviour-change plan (McEwan, 1998), the aim being that he or she becomes a fully autonomous learner (Westwood, 2003). Burgess (2003) notes that children with attention problems often blame external factors for their learning difficulties and that they need to form an internal locus of control. Thus, in an attempt to reduce impulsivity, children with ADHD are taught “stop, listen, think, say, do” techniques (McEwan, 1998; Rief, 1993) and are encouraged to become

self-monitoring and self-reporting (Burgess, 2003; McEwan, 1998) and even enter behavioural contracts (Rief, 1993).

Several other techniques are mentioned that can be used to help manage behaviour problems:

- Using baroque music (or any music with between 60 and 64 beats per minute) during times of self-activity (Bester, 2000)
- Ensuring consistent classroom routines (Gigout-Hues, 2006; Rief, 1993)

It is again important to note that often the strategies and techniques will be effective for only a while (Rief, 1993) and that they will have to be consistently reviewed and revised.

3.7.3 Organisational focus

A characteristic feature of children with ADHD is that they do not know how to organise themselves and the materials they will need to complete class work and homework assignments (Naparstek, 2002; Rief, 1993). Naparstek (2002) also notes that often these children really do want to do their school work, but are simply unable to complete it successfully.

Strategies that a teacher can employ include the following:

- Ensure that their workspace is organised and clear of unnecessary junk (Gigout-Hues, 2006; Rief, 1993).
- Check that the learners have the necessary requirements to complete the task (Naparstek, 2002).
- Write homework on the board instead of only giving it orally (McEwan, 1998; Rief, 1993).
- Assist learners with correctly writing down the homework (Naparstek, 2002; Rief, 1993).

- Divide longer assignments into smaller chunks with intermediate deadlines (Naparstek, 2002; “Working”, 2006).
- Provide frequent reminders of assignment due dates (McEwan, 1998).
- Use calendars to show future dates, thus assisting with long-term planning (McEwan, 1998).

Learners with ADHD should be encouraged to use a notebook or diary in which they can write down work that has to be done (Naparstek, 2002). Also, the use of different (or colour-coded) folders can assist these children with organising their work (McEwan, 1998; Rief, 1993).

3.7.4 Structural focus

There are structural classroom accommodations, which recognise the special needs of learners with ADHD, that can be used to positively influence the educational outcomes of these children (McEwan, 1998). The accommodations can be summarised in terms of making concessions and being flexible. Although Naparstek (2002) suggests changing the school environment once behaviour modification techniques and medication do not seem to be having an effect, such changes can well be seen as another element in the set of interventions that can be attempted for all learners who are struggling with attention problems.

One of the main accommodations that have to be made relate to the curriculum itself and the workload that goes along with it – and it can be summarised as emphasising quality rather than quantity (Gigout-Hues, 2006; McEwan, 1998; Naparstek, 2002). Calls have been made for modifying the curriculum for children with ADHD in such a way that school work is individualised for children’s specific needs (Burgess, 2003; McEwan, 1998; Rief, 1993). McEwan (1998) notes that learners with ADHD struggle particularly to keep up with the written work demands that are made on them and that the workload while seated should be reviewed and decreased where needed (Rief, 1993). Hallowell and Ratey (in Burgess, 2003) state the task as “simplify instruction, simplify choices, simplify scheduling”. As learners with ADHD struggle

particularly with written work, the use of word processors should be encouraged (Mayes et al., 2000; McEwan, 1998; Rief, 1993).

McEwan (1998) points out that the aim of schooling is to help learners learn and that to accomplish this without destroying their self-esteem and self-confidence may mean making adjustments to testing and marking schemes as well. The sorts of accommodations that can be used include the following (DuPaul & White, 2006; McEwan, 1998; Naparstek, 2002; Rief, 1993; Westwood, 2003):

- Experiment with different testing formats.
- Provide more time for tests.
- Permit breaks during tests.
- Allow the learner to dictate the answer to a scribe or onto audio tape.

Also, as seat work is particularly difficult for children with attention difficulties, concessions should be made that allow the learner with ADHD to play with a stress ball or similar object (Bester, 2000). Further, a teacher can permit movement while seat work is being completed and even provide short breaks as necessary (McEwan, 1998).

Along with seating learners with attention difficulties away from distractions, teachers can also allow such learners to use headphones while working, and even during tests, in an attempt to block out distractions (McEwan, 1998).

3.7.5 Medical focus

Medical interventions that are used will not be examined in any detail, as they are not something over which a teacher has much control, yet are included to give a fuller picture of the interventions that are usually used. Such interventions are also among the most controversial (“Attention Deficit Disorder”, 2006; McEwan, 1998).

Naparstek (2002) notes that in mild cases of ADHD, behaviour modification approaches may be sufficient. However, in more severe cases, medication may well

have to be included and has proven useful to many learners (McEwan, 1998; Naparstek, 2002). The medications used increase blood flow to the areas of the brain controlling alertness and attention, thereby improving functioning and impulse (McEwan, 1998; Naparstek, 2002). About 50% of diagnosed children in the age group 4-17 take stimulant medication ("Attention Deficit Disorder", 2006), and McEwan (1998) reports that more than 70% of children with ADHD who take medication have behavioural, academic, and attention improvements. It is believed to work against the negative spiral of poor academic and social performance and help children to believe in themselves (Bester, 2000). Authors note that the drugs do not control or change the child's basic personality or values and that children on medication maintain total freedom to choose how to behave (McEwan, 1998; Rief, 1993).

The drugs do, however, have short-term side effects, although they are usually not severe: stomach aches, headaches, irritability, tics, abnormal heart rate, increased blood pressure, loss of appetite, weight loss, and difficulty falling asleep (Naparstek, 2002; Rief, 1997; Szabo, 2006). It has been reported that 30% of children with ADHD do not respond to the stimulants or cannot tolerate the side effects ("Attention Deficit Disorder", 2006). Most doctors, however, feel that the side effects are outweighed by the positive psychological benefits of taking medications (Thompson, 2006).

The teacher's role is to help evaluate behaviour changes in response to differing dosages and differing medications (McEwan, 1998). It is important to note that the medications used do not cure ADHD (Bester, 2000). Also, it is important to remember that medication is only one of the interventions that are available (Rief, 1993) and that medication alone is not the answer to a learner's problems (McEwan, 1998).

3.8 Constructivism as an alternative

Constructivism is also offered as an alternative teaching approach for learners with ADHD. Constructivism, as noted earlier, tries to move away from pure skills (which

have become an end in themselves) and a lack of relevance in learners' lives (Harris & Graham, 1996a). Rather, the aim is to get children to participate in the learning, which will lead to deeper understanding of what is being learnt (Harris & Graham, 1996a).

However, some learners cannot cope with the high cognitive demands and task management requirements of constructivist classrooms, and they lapse into uncertainty, confusion, and even anxiety when faced with constructivist approaches (Loyens et al., 2006; Perkins, 1999). Also, it has been noted that learners who face learning challenges in terms of behaviour, among others, have greater difficulties in constructivist classrooms and that they benefit from more structured and explicit instruction (Harris & Graham, 1996a), as well as help with processes and strategies (Harris & Graham, 1996b).

The argument, then, is that explicit instruction has a place with some learners, and that the learner's special needs need to be taken into consideration when using alternative teaching strategies such as constructivism (Harris & Graham, 1996a; Harris & Graham, 1996b). The more pragmatic approach to teaching learners with special educational needs is to opt for an integrated instruction model, providing whatever level of support a learner may need, regardless of philosophy or paradigm (Harris & Graham, 1994; Harris & Graham, 1996a). It is such an integrated approach of using constructivist learning cycles that include direct instruction that will be attempted in this study and its effect on learners with ADHD observed.

3.9 Conclusion

The strategies that were detailed in this section were used to guide the presentation of learning material in the computer programming lessons used with learners displaying the behaviours associated with ADHD. It should be remembered that though these strategies can help children with ADHD learn, there are no foolproof methodologies or strategies that will work with all learners all of the time. ADHD children are "consistently inconsistent" (DuPaul & White, 2006, p. 60). There will always have to be constant reviewing and revising.

With the background provided by the literature in terms of (i) the use of constructivism in learning introductory computer programming and (ii) the range of strategies that can be used to enhance learning by ADHD learners, it is necessary to return to the question asked at the end of Chapter 2: is constructivism (as expressed in the Karplus learning cycle) one intervention that can be used to address the learning needs of learners with ADHD as they learn computer programming? Are the cognitive demands of constructivism too great for learners with ADHD, or can such learners successfully negotiate the construction of knowledge within carefully constructed social teaching environments based on a variety of constructivist learning approaches? Thus, the need for this study. How the study progressed will be explained in the next chapter.

Chapter 4 Research design and methodology

4.1 Introduction

In an attempt to produce a rigorous study that would be acceptable in both the computer science and education fields, this study follows the framework developed by Pears et al. (2002) for computer science education research (see Figure 3 for a simplified version of the framework). Such a framework should also help bridge the gap between learning theory and social science data analysis techniques (Pears et al., 2002).

In this chapter, the key elements of the Pears et al. (2002) framework are described, after which the implementation of the courses on which the study is based is detailed. Using the framework as guide, the design and methodology for this research project are then described. Finally, issues relating to the reliability and validity of the study, and of qualitative research in general, will be explored.

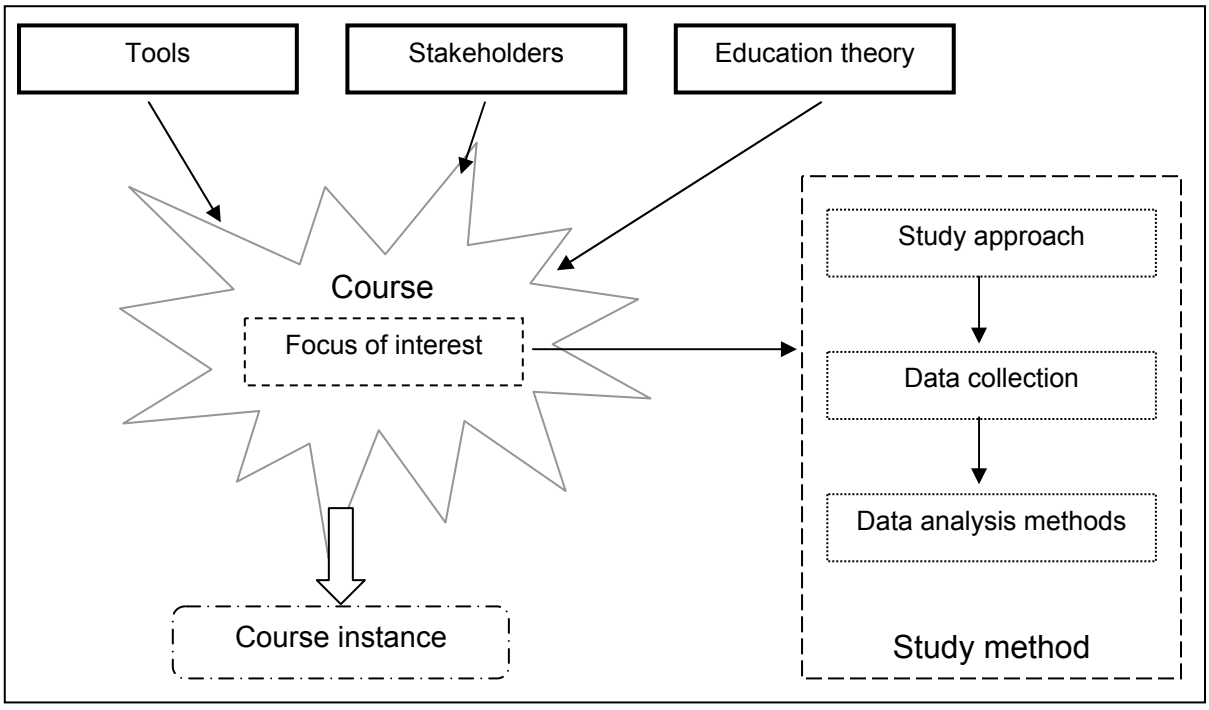

Figure 3 Research framework (after Pears et al., 2002, p. 102)

4.2 Research framework

There are two main categories in Pears et al.'s (2002) research framework: influences and evaluation/research.

I. The three influences that help define course content are:
   i. tools – these are the tools and technologies that will be used to implement aspects of the course;
   ii. stakeholders – this is the whole community that has an influence on the content, form, and approach taken in a course; and
   iii. education theory – represents the ideas (both implicit and explicit) that form the basis of the teaching approach.

II. Educational evaluation and research make up the second main category.
   i. Focus of interest – an aspect of the course about which more is to be learnt and without which there is no research.

   Researching the focus of interest leads to three other decisions that together define the study method:
   ii. Study approach – the investigative technique to be used, based on what is to be investigated.
   iii. Data collection – techniques that will provide relevant data.
   iv. Data analysis – the processes that will be used to extract knowledge and insights.

4.3 Course implementation

The research was carried out over two courses (that is, there were two course instances), the first taking place in 2006 and the second in 2007.

In the first course, the learning was centred on implementing a game. The computer would randomly choose a number between one and a hundred, and the user should be presented with a prompt to guess this number. Once the user has entered a number, the program should let the user know whether the guessed number is too

high, too low, or whether the number has been correctly guessed. Once the user gets the number, the program should also let the user know how many tries it took to guess the number. The user should then have the option of playing again or closing the program. The decision to use a game was based on the idea of a creative and interactive task that would appeal to learners with ADHD and would be able to hold their attention. There were 11 lessons of one hour each in the course that covered the basics of programming from variables to `If` statements and `While` and `For` loops. Some extras such as the use of text and background colour were included.

The second course started with some basic techniques in Delphi. The concepts associated with the use of forms and basic components (such as buttons, labels, and edit boxes) were covered. Some of the more common properties of these components were used as well, such as name, caption, font, visible, and tab order. The implementation code of buttons was used to change component properties. `If` statements were also introduced, for example when implementing the Close button to check that the user did mean to exit. Here we also used message boxes. Some conversions between integers and strings were also introduced, as well as the use of user-defined procedures. There were 12 lessons of one and a half hours each.

The second course continued with another game. The user draws “cards” from a pack of 13 cards (from ace to king) via the click of a button. These cards are then randomly chosen by the computer and displayed to the user. The aim of the game is to total the face values of the cards and get as close to a score of 21 without going over 21. The user then stops play at any point before reaching 21, and the computer stops the game if the accumulated score reaches, or goes over, 21. A new game can then be played or the game closed.

The details of the course, in terms of the tools used, the stakeholders involved, and the educational theory used, will be dealt with next.

4.4 Tools

The tools used to teach the introductory programming course ran on personal computers running Windows XP Professional in a networked computer laboratory (of 15 computers) in a secondary school. The network was simply used to share files and the single printer. Two integrated development environments were used:

- Dev-Pascal was used as a basic introduction to programming concepts. This compiler was used, as Pascal is a very simple language to learn and has English-like syntax. It was used to create simple DOS-based programs.
- Delphi was used to take the programming concepts further into a windowed environment. It has syntax similar to that of Pascal, and there should not be a huge jump from the one to the other.

In the first course, both the Dev-Pascal and Delphi development environments were used, the former for the first nine weeks and the latter for the last two weeks. The second course started directly with Delphi to see whether there was any difference in the learning of programming and in interest in the course.

4.5 Stakeholders

Three main sets of stakeholders were affected by this course. Firstly, as the course was run as an extramural activity, school management wanted to be sure that the activity would be worthwhile and properly managed. There was little concern about the content of the course as such, and as the activity was not part of the formal school timetable, there was no expectation of any assessment or reporting.

Secondly, learners in Grades 9 to 12 who wanted to learn to program were the main stakeholders. As this was a voluntary activity, there was no way of knowing in advance the number of learners who would be involved. Thus, there was also no way of knowing the spread of learners across the grades, nor the gender breakdown. No computer background was assumed.

As the activity took place at an independent secondary school for learners with special educational needs, the learners who became involved in the project had one or other specific learning disability and/or attention deficit/hyperactivity disorder. Although ADHD was often not diagnosed, the learners who attended the classes all showed the behaviours associated with ADHD as noted in Chapter 3.

The first group of learners was made up of three boys (one in Grade 10 and the other two in Grade 11). There were initially 10 interested boys, but most did not attend more than four of the sessions. In discussion with the learners, this appeared to have been for various reasons: (1) the classes clashed with afternoon extra lessons at the school; (2) learners were expecting more graphical/animated programming; and (3) it could also be that, as found in other studies (Robins et al., 2003), some learners had a reasonably accurate sense of how they were likely to fare within the first two weeks of a course and decided that this was not for them.

The second group was made up of four boys (two in Grade 9, one in Grade 10, and one in Grade 12). One of the Grade 9 boys left the school near the start of the course, and there were, thus, three boys for the main part of the course.

These small class sizes (along with the lack of control groups) meant that, as with other studies (Clement, 2004), it was difficult to use experimental approaches to determine whether the learners had learnt more. But then, Almstrum, Guzdial, Hazzan, and Petre (2005) have noted that it is sufficient to focus on small numbers of individuals in qualitative research. The small class size, however, was an advantage. Having fewer learners in a class made it easier to notice when a learner was heading off task, being distracted by other stimuli, or getting stuck on task. Thus, the task of using proximity control was made easier and helped keep the learners on task and progressing. A further advantage was that it was easier to pick up what all the learners were experiencing and doing and, thus, made a wider observation of classroom events possible.

The third stakeholder was the teacher who would participate in the study in the role of teacher-researcher.

As with any other research that involves human beings, the informed consent of the stakeholders has to be obtained (Baskerville & Wood-Harper, 1996; Padak & Padak, 2006). This means, further, that the researchers and participants in the study share a mutually acceptable ethical framework (Avison, Lau, Myers & Nielsen, 1999) – there are issues about what will be revealed in the study, to whom, and by whom.

All stakeholders were clearly briefed on the experimental and interventionist nature of the research. Permission to undertake the research was sought from both the principal of the school and the parents/guardians of the learners involved. These letters of permission can be found in Appendices A and B, respectively.

However, with the possibility of improved learning in the proposed research, it is probable that the participants would have welcomed the research (Baskerville & Wood-Harper, 1996).

4.6 Education theory

A constructivist teaching approach (discussed in detail in Chapter 2) was followed. It was further project- and activity-based with an emphasis on reflecting on the learning process (Jakovljevic et al., 2003). The role of the teacher-researcher was to plan appropriate learning experiences that would be relevant to the learners' needs and expectations and to use the strategies that are understood to assist learners with ADHD to learn.

In both courses, the general format of the lessons was much the same. The first lesson covered basic computer architecture to provide a mental model on which to build the rest of the course. Thereafter, the Karplus learning cycle referred to in section 2.5 (exploration, concept introduction, and concept application) was used to introduce topics: (1) individual learners would be given a running program (in executable form) and would be given an opportunity to play around with it to see what it did and how it functioned; (2) learners would be divided into pairs to work their way through the printed code to begin to understand the function of the code itself, with direct instruction included where necessary; thereafter, a whole-class

session would be held to discuss and agree on the function (and syntax) of the code; (3) the newly learnt concept would then be applied individually in practice programs or the game that was implemented over the development of the first course, with help from peers and the teacher as required.

Although this basic pattern was followed, the exact detail and plan of each lesson were different in an attempt to find a methodology that worked best at facilitating learning. The linear nature of Pascal code made step 2 above easier than the event-driven code of Delphi. Thus, in the second course, stepping through a program or demonstrating its creation was attempted in Delphi. When demonstrating the program creation, the process was chunked into manageable pieces and demonstrated portion by portion, allowing the learners to implement the sections as the demonstration progressed. On occasions, the concept introduction was handled as direct instruction, where the concept (such as an `If` decision structure) was diagrammed on the board with the code being written next to it.

Further constructivist techniques were applied as required:

- When designing programs, the tasks needed would be written on cards (paper first) and then arranged on a table in the correct order using the paper "tasks" as manipulatives.
- Program code was scaffolded with code segments and comments to provide learners with something to work from.
- Socratic questioning was used when questions arose. Learners were always encouraged to justify their ideas and solutions to the class and were also expected to agree/disagree with another learner's ideas and give reasons for doing so.

Strategies for assisting ADHD learners were also kept in mind. The demonstration offered by a working program helped the learners see what could be done and, as such, also provided a completed example of what could be expected. Chunking the programming tasks into smaller sections allowed for learners with attentional and working memory limitations. Further, instructions, provided in typed format as a

guide, laid out the steps to be taken in sequential format and were a form of chunking, so as not to overwhelm the learners. Such instructions also allowed the learners to set up directories, so assisting with organisational problems. Questioning, using Socratic techniques, allowed for active participation by learners. Further, ample wait time between question and answer, for learners to formulate and organise an answer, was allowed. Repeated pointing out of compiler error messages assisted learners in developing information-seeking strategies so that they could begin to find information for themselves.

Having examined the learning environment (tools, stakeholders, and learning theory), the focus will now shift to the research environment (the research question, study approach, and data collection and analysis).

4.7 Focus of interest

The aim of this research was to study the use of constructivism, in the form of the Karplus learning cycle, in teaching computer programming to high school learners with ADHD, thereby determining

1. what constructivism is and how it can be used to support the teaching and learning of programming (Chapter 2);
2. what the Karplus learning cycle is and how it can be used to support the teaching and learning of computer programming (section 2.5);
3. what ADHD is and what the strategies are that can be used to promote learning by children with ADHD (Chapter 3); and
4. whether the Karplus constructivist learning cycle is successful in the teaching of computer programming to children with ADHD?

It remains now to examine whether the Karplus constructivist learning cycle can be successful in the teaching of computer programming to children with ADHD.

4.8 Study approach

Myers (1997) notes that there are two main research methods: qualitative and quantitative. Quantitative studies originated in the natural sciences and include methods such as surveys, laboratory experiments, and numerical methods and modelling (Myers, 1997). Such research is aimed at determining cause and effect with a view to prediction and is based on confirming or rejecting specific hypotheses that are formulated at the start of the study (Hazzan et al., 2006; Hoepfl, 1997). Qualitative studies, on the other hand, come from the social sciences and use methods such as interviews, documents, and participant observation (Myers, 1997). The aim in these studies is rather to enlighten, understand, and explain in a coherent manner (Hazzan et al., 2006; Hoepfl, 1997). A useful analogy is given by Patton (in Hazzan et al., 2006): both quantitative research and qualitative research result in images, but quantitative studies produce photographs that capture and freeze moments in time, whereas qualitative studies produce a film that gives a sense of movement and development over time. The choice between the two, then, is one of objective (Hazzan et al., 2006). Further, these approaches are not mutually exclusive, and research studies could use elements of both approaches (Hazzan et al., 2006).

This study follows a qualitative approach, and as such, it is research that "produces findings not arrived at by means of statistical procedures or other means of quantification" (Strauss & Corbin in Hoepfl, 1997).

4.8.1 Research paradigm

Klein and Myers (1999) further note, based on Chua (1986), that there are three basic paradigms of qualitative research: positivist, interpretive, and critical. This study is an interpretive one where it is assumed that understanding is achieved through social constructions such as language and shared meaning, accepting, therefore, that there may be multiple realities (De Villiers, 2005). In the context of teaching computer programming, this would mean attempting to understand the learning of computer programming through the meanings that learners assign to the

syntax and objects used in the implementation of a program and through the language with which they convey their understanding. The aim is, thus, to make sense of, and understand, the context of learning to program constructively in an ADHD classroom and the process whereby learning is influenced by, and influences, this context. (This meaning has been adapted from the aim of interpretivist studies in information systems as given in Myers (1994) and Myers (1997).)

4.8.2 Methodology

The methodology used in the research was action research. The study was, thus, an empirical one based on primary (or new) data, leading to textual (rather than numerical) data, in an environment with a low degree of control (that is, in a natural setting) (Mouton, 2001; Myers, 1997). Although there is not a rich background of empirical research in computer science (Clancy et al., 2001; Hazzan et al., 2006), computer science education research is an inherently transdisciplinary field (Clear, 2001), and there are examples of action research and other qualitative approaches being used to study learning in computer science (Clement, 2004; Hazzan et al., 2006). Further, authors point to the fact that formal experimental methods in exploring, or describing the complexity of, teaching methods are of doubtful value and inadequate for understanding human action (Almstrum et al., 2005; Clear, 2001; Doolin, 1998). However, qualitative methods such as action research are used for the investigation of social phenomena and are certainly suitable, and gaining acceptance, for the study of applied fields such as learning and teaching practice where it has had some influence (Abell, 2005; Clear, 2001; Feldman, 1994; Hazzan et al., 2006; Mouton, 2001; Myers, 1997; Sommer, 1987). There is, thus, the recognition of human actors in the research rather than human factors (Myers, 1994).

Action research, a distinctive method since the 1940s (Kock, Avison, Baskerville, Myers & Wood-Harper, 1999; Kock, McQueen & Scott, 2006), is not always clearly defined, and there are a variety of forms, traditions, and emphases (Feldman & Minstrell, 2000; Hult & Lennung, 1980; Newman, 2000). Action research has been described as a

> participative, practitioner-researcher approach [that] lends itself to the domain of educational research, where an evolving intervention or product is investigated over several cycles (De Villiers, 2005, p. 146).

It has been noted that a research environment can be more deeply understood if the researcher is part of the environment (Kock et al., 1999), and action research takes this a step further by encouraging the researcher to experiment through intervention and then reflect on the effects of that intervention (Avison et al., 1999). Thus, action research merges research and praxis, theory and practice, and thinking and doing, having both action outcomes and research outcomes, leading to relevant research findings (Avison et al., 1999; Baskerville & Wood-Harper, 1996; De Villiers, 2005). The role of the teacher can, thus, become that of researcher, and vice versa.

Action research was initially envisaged as a two-step process: (1) diagnosis, followed by (2) a therapeutic phase (Blum, 1955). However, some additional structure has been imposed to achieve a more rigorous approach. Action research is, thus, now seen as a five-phase cyclical process rather than as an event (Baskerville & Wood-Harper, 1996; De Villiers, 2005; Jakovljevic et al., 2003; Kock et al., 2006):

1. Diagnosing – identification of initial problem.
2. Action planning – specifying actions that should relieve or solve the problem.
3. Action taking – implementing planned and appropriate actions.
4. Evaluating – evaluating the outcomes of the instructional action.
5. Specifying learning – an ongoing process that restructures the instruction in light of what was learnt.

The cycle then continues (whether the intervention was successful or not) and so adds to the theory being built up (Avison et al., 1999; Baskerville & Wood-Harper, 1996), relying on the cyclic nature to achieve internal consistency and validity (Kock et al., 2006). Action research is, thus, "empirical, yet interpretive. It is experimental, yet multivariate. It is observational, yet interventionist" (Baskerville & Wood-Harper, 1996, p. 236). Figure 4 is a representation of the action research model that depicts its spiral nature closing in on a solution, with the researcher playing a central role.

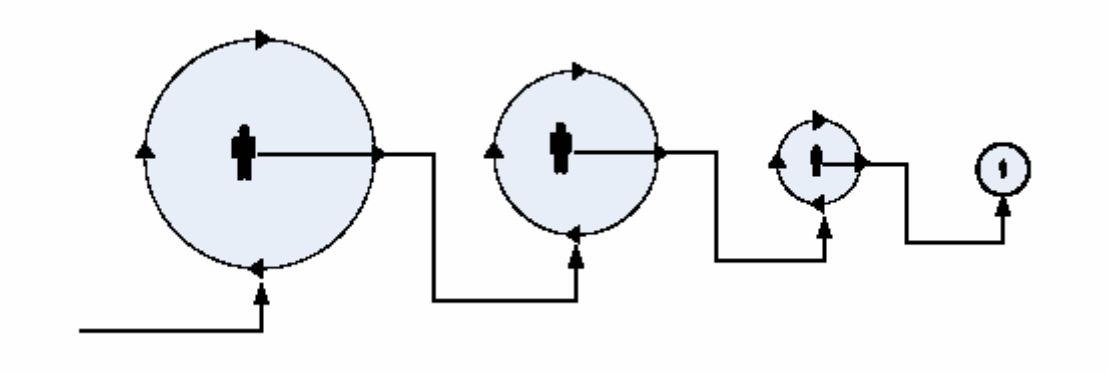

Figure 4 An action research model (after De Villiers, 2005, p. 147)

As a teacher attempting to find the best way to teach learners with ADHD introductory programming, the classroom is the ideal environment in which to use action research (Baskerville & Wood-Harper, 1996). Further, the knowledge that is obtained is immediately applied and the learning strategies applied in a cyclic process. Also, the action research approach should give access to the internal thinking processes of the learners and should, thus, be more helpful than simple measures of performance (Ben-Ari, 1998). This would be in agreement with Ben-Ari et al. (2004, p. 230) who note that "[t]he only way to understand and improve the complex field of Computer Science Education is to be able to see the principles, the concepts, the skills, the necessary competences and capabilities of the computer scientist from the perspectives of the learners". The aim is, then, to improve and understand (Feldman & Minstrell, 2000) the learning that happens in the classroom.

Action research is not without its challenges, some of which are noted below:

- As the approach is inherently non-reproducible and context-bound (Baskerville & Wood-Harper, 1996; Feldman & Minstrell, 2000), there can be issues surrounding its external validity (Kock et al., 2006).
- Low control is associated with natural settings (Kock et al., 2006).
- Dual goals, namely, to improve learning (pragmatic) and generate knowledge (scientific), can lead to goal confusion (Kock et al., 2006; Rapoport, 1970; Williamson & Prosser, 2002).

- The level of closeness of the teacher-researcher relationship can lead to researcher bias (Baskerville & Wood-Harper, 1996; Feldman & Minstrell, 2000; Kock et al., 2006; Rapoport, 1970).
- As action research is a journey with an unplanned and informal structure, it can make informed consent a little less informed (Kock et al., 2006; Williamson & Prosser, 2002).

These problems, however, are not unique to action research and are general problems with the interpretive research methods of social science (Baskerville & Wood-Harper, 1996).

The objective of simultaneously investigating and changing instructional strategies and techniques would be to enhance the learning of introductory computer programming by children with ADHD, including the flexible use of good practice (Feldman, 1994). Although interventions and adjustments were made to the teaching and learning methodology as the two courses progressed, one full cycle of action research was completed during the second course just before starting the 21 game (see section 4.3). Evaluating, specifying learning, and diagnosing (the latter two phases of the first action research cycle and the initial phase of a second action research cycle) then took place. Further action planning and action taking would be implemented during the second course as the implementation of the 21 game progressed. The results reported here focus on the first action research cycle that took in the whole first course and the start of the second course.

Each qualitative research methodology will use several techniques for collecting empirical data or materials (Myers, 1997). For the action research study detailed above, the techniques described in the following section were used.

4.9 Data collection

The study was an action research project that used qualitative data sources. Data triangulation was used by collecting data from several different sources around the same learning event, allowing each to complete, deepen, and broaden the findings

made in the others and ensuring that variances were not a result of the data collection method (Feldman & Minstrell, 2000; Hazzan et al., 2006; Jick, 1979). These will be explored below.

4.9.1 Observation and diary writing

The teacher-researcher noted observations, behaviours, and reactions in the classroom via diary writing. The text then became a narrative that documented what people said and did over the course of the research (Myers, 1994). Such techniques have been used in (and suggested for) action research in other studies (Abell, 2005; Baskerville & Wood-Harper, 1996; Duveskog, Sutinen, Tedre & Vesisenaho, 2003; Hazzan et al., 2006; Sá, 2002). Brief notes and keyword reminders were noted during class time and detailed diaries written up immediately after each instructional event, recognising that the quality of the diary would depend heavily on the memory of the teacher-researcher (Hughes, 1996; Sá, 2002). In a sense, then, the diary represents a "textual snapshot" (Davis et al., 1992, p. 303) of the events in the learning environment. Such diaries also helped to increase the level of reflection on the part of the teacher-researcher (Sankaran, 1997).

4.9.2 Interviewing

A formal, semi-structured interview was held with the learners as a group towards the end of the first course as part of the evaluation phase (Hoepfl, 1997). Some questions were decided beforehand, but the interview was an open-ended one. Field notes were taken in this interview. Also, informal discussions were held with all learners (individually and in groups) as the research progressed. Such interviews were recorded as part of the diaries. The interviews gave the researcher an opportunity to learn about the learners' thinking processes and to look for the origins of what was seen in the observations (Hazzan et al., 2006). Further, interviews gave the researcher an opportunity to get participant feedback on the interpretations that had already been made (Feldman, 1994; Williamson & Prosser, 2002).

4.9.3 Documents

Learners' program code was printed for some of the tasks and kept for later analysis (as done in Duveskog et al., 2003). This provided a view into the logic and style of coding being used by the learners.

Davis et al. (1992) note that, in interpretive studies, observation guides interpretation as much as interpretation guides observation. There is, thus, not always the clear distinction between data collection and analysis that may be found in quantitative studies (Myers, 1997). The following section deals with the modes of analysis that were carried out in the study.

4.10 Data analysis

Grounded theory has been defined as "an inductive, theory discovery methodology that allows the researcher to develop a theoretical account of the general features of a topic while simultaneously grounding the account in empirical observations or data" (Martin & Turner, 1986, p. 141). While grounded theory is usually used as a research methodology, it can also be used as a mode of analysis for qualitative research (Martin & Turner, 1986; Myers, 1997) and, as such, deals with the cognitive problems associated with analysing qualitative material by bringing them out into the open (Turner, 1983).

In this study, a grounded action research approach was used, merging the analysis techniques of grounded theory within the process of action research, leading to improved rigour in the inductive theory development (Baskerville & Pries-Heje, 1999). The grounded theory approach aims to develop theory that is grounded in the contextual data collected, and there is a continuous interplay between data collection and analysis (De Villiers, 2005; Myers, 1997).

The three coding procedures of grounded theory – open, axial, and selective coding – were used to analyse the textual data in an attempt to reach a deeper understanding of the teaching and learning environment being researched

(Baskerville & Pries-Heje, 1999; Corbin & Strauss, 1990; De Villiers, 2005; Pandit, 1996). In open coding, concepts in the data are identified and named and represent an important idea in the data (Baskerville & Pries-Heje, 1999). These form the basic units of the analysis (Pandit, 1996). Concepts are grouped, or categorised, at a more abstract level into categories, bringing similar concepts or ideas together. Axial coding is then used to find relationships between the various identified categories, and finally, a story (or core category) is developed that identifies the central phenomenon in the data (selective coding). This process is shown in Figure 5. Memos were written as the process progressed, and these served to reflect on the concepts that had been identified and to integrate them into categories (Sankaran, 1997). Thus, the theory emerges systematically and inductively, integrating new concepts into the theory, with reviews and revisions where necessary. It does need to be remembered, though, that any interpretation is embedded in the researcher's cultural history and that all judgements are based on tacit theories, values, and beliefs (Newman, 2000).

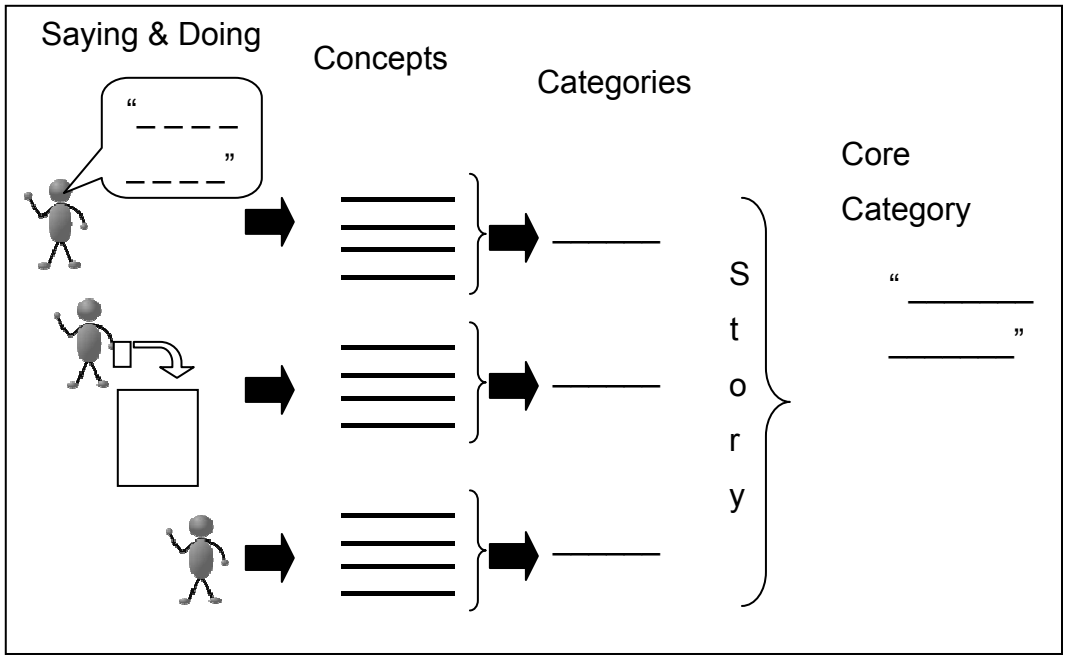

Figure 5 An overview of the grounded theory coding process (after Baskerville & Pries-Heje, 1999, p. 6)

However, action research and grounded theory cannot be fully integrated, as action research is too interventionist and goal directed (Baskerville & Pries-Heje, 1999). As the point of the research was to examine the effectiveness of the use of learning

cycles in the teaching of introductory programming, the core category of the learning cycle would have been defined at the start of the research, and the research would, thus, have been about exploring this category in more detail. Grounded theory can be integrated into action research, though, through the use of grounded theory notation and the use of the coding process to analyse the qualitative data that the action research process has generated. The coding was used during the diagnosing, evaluating, and specifying learning phases of the action research cycle as the data was being analysed and the results fed into the action planning and action taking phases of the action research cycle.

As the action research progressed, these coding procedures overlapped and were, thus, not done strictly sequentially (Baskerville & Pries-Heje, 1999). Further, within each cycle, new categories might appear that would lead to revised axial (and even selective) coding. In terms of this understanding, "action research cycles reach a termination point when the categories reach saturation. This means the evaluating and learning phases produce little change to any of the categories, especially the core category" (Baskerville & Pries-Heje, 1999, p. 8). Although this is the ideal, there was not time in this study (with only one cycle) to reach saturation point.

Grounded theory analysis is based on the contextual and process data collected (Orlikowski, 1993). Such multiple sources of data (as indicated in section 4.9 above) not only provide multiple perspectives on the educational environment (Orlikowski, 1993), but also enhance validity and reliability (Pandit, 1996). It is to these two concepts that attention now shifts.

4.11 Reliability and validity

Reliability refers to the extent to which research, when repeated, yields the same results and validity to the degree to which research accurately reflects the concept it is supposed to measure ("Writing @ CSU", 2005). Lincoln and Guba (in Hoepfl, 1997) have questioned these categories and have suggested an alternative set of criteria that better accounts for the realities of qualitative research:

- *Dependability replaces reliability*. It has been suggested that an "inquiry audit" (where reviewers examine the process and product of a research process for consistency) will enhance the dependability of a research project (Hoepfl, 1997). As action research involves a unique intervention in a particular situation, there is no way that it can ever be repeated, and so dependability will have to depend on a synchronic reliability via triangulation (Baskerville & Wood-Harper, 1996). As there will only be one coder (the teacher-researcher), some of the stability and reproducibility issues ("Conducting Content Analysis", 2006) will reduce to consistency issues. Reviewing earlier analysis regularly can help overcome this concern. Further, ensuring that all data is available for other researchers to review (without violating confidentiality issues) can help to ensure that all researchers will arrive at the same conclusions (Baskerville & Wood-Harper, 1996; McKnight, Magid, Murphy & McKnight, 2000). The teacher-researcher needs to recognise that an isolated researcher can lead to bias (McKnight et al., 2000).
- *Transferability replaces external validity/generalisability*. The highly contextual and authentic nature of qualitative research, and the intentionally interventionist approach of action research, mean that results are not generalisable. Rather, with sufficient context detail, other researchers can determine the degree of similarity, judge the relative value for themselves, and transfer results as appropriate (Hazzan et al., 2006; Hoepfl, 1997).
- *Credibility replaces internal validity*. The issue here is the extent to which results accurately describe reality or, rather, adequately represent a part of reality (assuming multiple realities) ("Conducting Content Analysis", 2006; Hoepfl, 1997) – have facts, actions, and actors been appropriately connected in the natural context (Sá, 2002)? Credibility is, thus, dependent on the richness of the data gathered and the analytical ability of the researcher to make clear and meaningful links between the data and conclusions (Hoepfl, 1997; McKnight et al., 2000). Further, participant observation (as in action research) allows for more candid information exchange and a richer data source (Kock et al., 1999). It can be enhanced through triangulation (Hoepfl, 1997; McKnight et al., 2000; Sá, 2002): various sources of data pointing to the same conclusions, as well as getting feedback from participants to validate conclusions.

There is a perspective that generalisability is irrelevant for educational situations (Hazzan et al., 2006), as each one would be unique in any case, and that as understanding is the primary focus, there is no need to show that a particular result is valid for all situations (Feldman & Minstrell, 2000). However, criticisms of action research by positivists can also be used by practitioners of action research to look carefully at their approach and re-evaluate it (Kock et al., 2006). It has been argued (Kock et al., 2006) that such a review (especially of the cyclic nature of action research) may show that many of the criticisms of action research may not be justified and that the "*effective application of the iterative approach to action research has the potential to bring research rigour up closer to standards acceptable by positivists*" (Kock et al., 2006, n.p., italics in original).

4.12 Conclusion

This chapter presented the research design and methodology for this research. The Pears et al. (2002) framework was used to ensure a rigorous study that addressed all the aspects necessary to understand the complex educational setting and so improve both teaching and learning. The tools used and stakeholders involved were spelt out, as well as the constructivist education theory that was used. Constructivist teaching approaches were applied during the two course implementations described, in lesson formats based largely on the Karplus learning cycle, keeping teaching strategies for learners with ADHD in mind.

The focus of the research was presented as the use of constructivist approaches, in the form of the Karplus learning cycle, in teaching introductory computer programming to high school learners with ADHD. To achieve this, a qualitative and interpretive approach was used using an action research methodology to investigate the process, and data was collected and analysed using grounded theory methods. Finally, understanding the validity and reliability of such a study in alternative terms was explored.

The findings of using the Karplus learning cycle to teach introductory computer programming to learners with ADHD will be presented next. It will be seen that this three-phase cycle needs to be extended slightly in order to facilitate such learning.

Chapter 5 Findings and discussion

5.1 Introduction

As noted in the discussion of action research section above (section 4.8.2), there are five phases to the action research cycle: diagnosing, action planning, action taking, evaluating, and specifying learning. The question that prompted this study (to what extent can constructivist methods be used to teach learners with ADHD introductory programming?) makes up the first phase, diagnosing. From here, action planning involved planning an approach to teaching introductory programming using the Karplus learning cycle (section 2.5) and other constructivist learning methods (section 2.4), bearing in mind that there are techniques that have been shown to assist learners with ADHD to learn (section 3.7). Action was then taken in the implementation of the two courses on which this research is based (section 4.3). Data was collected (section 4.9) while the action taking phase was implemented.

The process and responses must now be evaluated and learning specified. The grounded theory coding process is the essence of the 'evaluating' and 'specifying learning' phases (Baskerville & Pries-Heje, 1999), and this then leads to further diagnosing and the start of the next action research cycle.

Data analysis involved the identification of emerging patterns in the learning and teaching processes. These patterns were then translated into concepts that were further classified into four overarching categories. In this chapter, the initial concepts and categories are identified and the core category explored. This will lead to diagnosing of problems that became evident in the learning process and planning action in terms of expanding the Karplus learning cycle.

In the sections that follow, where quotes have been taken directly out of the field diary, they will be referenced by page and paragraph number in square brackets: [page-paragraph]. Where learners are referred to directly, the learner initials that were used in the diaries have been replaced by numbers: L1, L2, and so on. Although there were only three and four boys who attended the two courses

(respectively) on a regular basis, some other learners did attend the after-school classes, and so reference will be made to more than just seven learners.

5.2 Initial concepts

In the 'evaluation' and 'specifying learning' phases of the first action research cycle, open coding identified 22 concepts in the research diaries (that included notes on lesson observation, formal and informal interviews, and notes from learners' program code). I now discuss each of these concepts in turn, referring to evidence from the data.

5.2.1 Specific and direct questioning (C1)

Questioning must be targeted, as vague questions lead to shotgun answers – that is, answers all over the place. For example, when looking at computer adverts to get an idea of what computer hardware is required to run a program, the teacher "forgot to mention looking only at hardware, and so some software answers were given, too" [1-1]. Vague questioning also led to vague answers when looking at what a running program achieves: "... gave answers on what it does at a very general level – 'It says hello'. After some questioning, we got to: it asks for name, it reads name, it prints greeting, waits for enter, ends" [2-1].

Thus, questions must be clear and specific. If an answer at a high level of abstraction is sought, the question must be at that level. If the aim is to find out from the learner exactly what detailed steps occur or what the outcome of a specific action is, the question must be phrased in a way that elicits an appropriate response.

5.2.2 Level of detail of learners' questions (C2)

When the teacher asked questions that were not well formulated, learners responded with irrelevant questions that showed that they did not know exactly what was required. For example, while looking for the main hardware components in exploring computer architecture, it led to questions such as "What does 2.8 GHz

mean?" [1-1]. This, then, in a sense, follows on from the first concept (section 5.2.1 above) – not only did poorly thought out questions lead to poor answers, but they led to learner questions that indicated their lack of understanding of what the teacher had intended to ask.

5.2.3 Independent use of mastered concepts (C3)

Learners grasp concepts by seeing multiple examples and through repetition. For example, I needed to get the idea across that a programmer needs to tell the computer everything that it has to do. Thus, I asked the learners what steps needed to be taken when making a cup of tea: "Once they got the idea of thinking in greater detail (open lid, check water level, if not enough, get jug, ...), the rest flowed quite easily" [1-3].

This concept was also seen when doing trace tables: "The Grade 11s were beginning to see what was happening (after two loops through the while) – Oh, I now see what is happening" [8-1]. This type of learning by repetition was also found in the saving of work in folders when working with Delphi: "I reminded them to create a new directory – and L1, L5, and L6 noted: and save all files in it straight away – repetition is doing its thing" [30-2].

Thus, once understanding has been constructed, the concept can be used independently by the learner. That examples and repetition lead to independent learning was also evident when creating forms in Delphi, as well as when writing code:

- "no problems experienced in the layout of the form and the choice of controls" [30-3]; and
- "when a new approach was learnt, it was used: `.Show` instead of `.Visible:=True`" [21-3].

5.2.4 Variation in coping with the level of abstraction expected (C4)

Answers to the teacher's questions tended to be at a very general level:

- "Let them run the program executable and describe what it does. Responses were very general – draws line of stars. Had to ask them to describe output to get that a star is printed, pause, and so on till 10 stars shown" [6-1]; and
- "L9 caught on quickly ... Started with pen and paper exercise – how add two numbers? What info needed? How get it? I had expected the boys to do better, but I needed to talk them through virtually all of it" [3-4].

While some learners did get the level of abstraction that was expected, others did not. Returning to the exercise of listing the steps required to make a cup of tea, it was noted that "some grasped quite quickly the level of detail required. Others seemed stuck on the gross actions" [1-3].

Questioning can best be used to guide the level of detail required, as noted above (C1). The question remains, however – does one start at a more overall or general view of what is required or a more detailed level of abstraction? I do not think there is one answer to this question and that the approach will probably be determined by the circumstances and what it is the teacher is trying to achieve. As the learning cycle structure was the main thrust of this study, this question was not investigated any further and would make an interesting future development of the current study.

5.2.5 Using the learners' current knowledge in Socratic questioning (C5)

Socratic questioning can be used in various situations and uses the current levels of learner understanding (knowledge that is already available) to guide the learner to where an answer may be found. Situations where Socratic questioning was used successfully showed that it could be used in all areas of code development:

- What to include in the design of a user interface ("Via Socratic questioning, I got what was needed: heading, instruction, spin edit (or edit), %, button, and result" [29-4])

- In the use of a trace table to understand the purpose and flow of a section of code ([7-1])
- When a program was not exiting correctly ("I asked questions to get them to realise they were not changing the value that the loop was using to control entry to the loop" [10-1])
- In the value or use of hidden close buttons ("Via Socratic reasoning, saw that it is not a good idea because the user might want to close without changing the form purple" [20-2])
- In the use of component properties ("Occasionally, they struggled with which property to go for ... used the Socratic method to direct their attention to the properties box and to look for the property names there" [21-2])
- Where to place code in a program ("L1 ... needed help with where to put the code (via Socratic questioning)" [24-2])

Socratic questioning can also be used to help break an impasse and allow the learner to proceed with coding: "I guided him (L6) here via Socratic questioning, and he was able to answer and move forward" [34-2].

In each of these situations, the questions pointed the learners to something that was on the screen or something that they already knew and guided them in using the information to move forward with their programming plans. This is important for learners who struggle with attention problems and not because they struggle to focus their attention on where the problem is. Rather, there is so much to focus attention on that the one thing that can help them move forward is not seen – the focus of attention is constantly shifting between all that is on the screen.

5.2.6 Giving learners time to voice their understandings (C6)

Often, time was not given for learners to express their understanding: "I made a mistake here and tended to give answers rather than seeking further (using the Socratic method). Also, I did not give children time to say why they thought the user was in complete control" [1-4]. When introducing loops, "instead of asking them what could be done about this, I showed them the `For` loop in my program" [6-2]. I

would ask a question, get an answer, and then would lead the discussion without getting the learners to express themselves: "I should have asked the class to comment – again, I led the discussion" [8-3].

Allowing learners time to express their opinions is vital in constructivist approaches to learning, as it allows learners to explain why they are doing something, so revealing their underlying beliefs. Also, it encourages learners to use the language of programming in their explanations. Simply giving answers to learner questions means that the teacher misses an opportunity to see what the learner already understands: considering the placement of `Var` statements (globally as compared to locally), "he (L6) added it just below `Implementation`. When I pointed out the discrepancy with the printed code he had ... he asked if there was a difference. I should have got the whole class around. Instead I tried to think of an answer (as the code would run regardless) and then talked a little about scope (without using the word 'scope'). L6 moved the `Var` statement into the procedure" [37-1].

The reasoning behind giving answers could be that the teacher is afraid that the learner's answer may be incorrect and that this answer may confuse others. When developing the function using a trace table, "their expressions showed that they were getting the idea (except maybe L2 and L3). Instead of my asking what they were seeing (and checking whether they were predicting correctly), I was afraid they might be wrong, said that they were getting the idea (although I wasn't entirely sure what that idea was), and rather than have them confuse the rest of the class, continued on with the trace table" [8-1].

Also, the terminology required to express this understanding will come in time. "They did see that when a user enters a 0, the code skips to after the loop. I tried to change the terminology to 'continues after the loop, but skip was used again later/soon after. ... I used the terminology of how many times the loop had been executed, but realise now that their terminology was probably better" [8-2].

It takes courage to allow time for learners to express their understandings. There were occasions when time was allowed, and learners did come to some new

understandings on their own: when discussing the use of `Var` in function arguments, ”the understanding that was generally agreed on was that it was there for values that the main program did not yet have a value for” [13-2].

5.2.7 The value of McDermott's cycle (C7)

The McDermott cycle is the predict-confront-resolve cycle discussed in section 2.4.9. The compiler provides the perfect environment in which to use this learning cycle. A typical usage can be seen in the use of the square brackets after a string variable (for example, `string[20]`): “L1 asked what the `[20]` meant, and what would happen if the name was longer. Told him to try it out. He did and realised you could type in more characters, but that only 20 were displayed” [2-2]. Also, “I found it easier not to simply give answers, but to let them change values/operators for themselves and see the effect” [5-3].

The debug ability of a compiler further allows learners to step through a program and predict where the execution will jump to next and, when actually proceeding to the next step, see whether their prediction was correct: “Got everyone around L1's PC and stepped through the program, getting them to predict whether the code in the `If` would be executed. They then saw why the result appeared as it did. L1 then realised `Else` was needed” [31-2].

This cycle is also useful when learners believe that they have implemented a segment of code correctly, thus tacitly predicting that the program will run. The compilation process then becomes an opportunity to confront any errors, and these errors need to be resolved. The use of such compiler error messages helps learners to pick up their own errors and learn from the common mistakes that they make: “Used compiler error messages ... to help from `total:=total + spdNumber.Value` to the need to declare the variable first with `Var total: integer;`” [35-1].

Not only did the learners “seem to learn what things do/mean quicker this way” [5-3], but the use of this cycle allowed for the “I see it now” experience: “L4 wanted to

know what `answer:0:2` meant. I told him to change its value and see the response/effect. A little later he said – I see it now – when he realised that it specified width" [4-4].

5.2.8 Reliance on the more knowledgeable learner (C8)

The relationship between learners is often based on knowledge (and, thus, authority), although at first this may appear to be based on age:

- In class discussions, "L1 did most of the talking while L3 stood at the board" [17-2].
- "Learners tended to follow/believe the older boy who spoke with more authority" [2-3].

The learners who sat back then only participated in class discussion when asked direct questions: "this pair struggled and sat listening most of the time" [7-4].

The more knowledgeable learners see things faster, and the others then follow their lead. At times, such a learner may realise this is happening and try to work against it: "I noted that L1 (with more experience than the other three) often hung back and did not immediately give answers" [30-1].

5.2.9 Variations in the Karplus cycle (C9)

Initially, this cycle was used as follows: run the program, review the code, and then practise the concept in a new program. Learners "seemed to have a fairly good understanding of what each line meant" [2-3] where simple Pascal code was involved.

I also tried looking at the Pascal code first (as the exploration phase of the learning cycle), but "they did not realise the significance of the `While-Begin-End` structure" and "they also could not make sense of the use of `count`" [7-4] until a trace table was used to step through the code. I also tried this with a "partially completed

program" [12-3], where they used program and variable names to infer the function of the program.

Which of these two (program first or code first) worked best for learners did not lead to a universal answer: "L1 felt that reading code on paper first was better – gave him a chance to go through the program like a computer does, from top to bottom, to get an understanding of it. L2 felt seeing how it ran helped. They agreed that a combination of the two is better" [15-2].

This approach also meant that code structures were only introduced in the concept introduction phase of the cycle when needed in an application: for example, "showing them the `delay` got them to the solution" [6-1] in the case when stars were to be displayed on the screen one at a time. Learners "felt that learning structures as they needed them was good, as ideas came up when they needed to use them. They felt it helped them remember how to use them" [15-2].

This approach was also tried with Delphi programs. However, the reviewing code part often did not lead to understanding, as the Delphi code lost learners in all the automatically inserted declarations: I found that "terms like `Interface`/ `Implementation` were too big for them to figure out ... I am not convinced the exercise was worth it – there did not seem to be the 'aaahs' that indicate understanding" [18-6]. Reviewing code had very bad results on some occasions: "When I pulled out the code to show them, L6 glazed over immediately" [36-3].

Another variation of the cycle was tried: "Started today via demonstration of functioning code ... looked at the code and discussed design time/run time property selection/setting. Then they went off to their PCs ... to implement" [19-2]. The result was that "most seemed to cope" [19-3], although this did not work for all learners. The middle phase was sometimes supplemented by "diagramming on the board and wrote the code next to it" [23-1], resulting in "lots of blank looks – I am going to have to find another way of presenting this" [23-2].

It needs to be mentioned that, on some occasions, the learning cycle simply did not work at all: "L1 participated, as did L5, but L8 glazed over, and L6 was very distracted and took no interest – answered no questions, asked no questions" [32-40]. These instances may have been situations where, when working with ADHD learners, an approach works fine some days and not at all on other days.

5.2.10 Non-intuitive code syntax (C10)

Some code syntax concepts did not come easily and were a cause for confusion. Typical examples include the following:

- "Common problem ... – `write(' ... ');` and `readln('a');`" [5-5] where the use of inverted commas was not properly grasped.
- "... got it going with some errors (the usual ones: '`:=`' and '`=`' confusion; '`;`' left off at end of line; no `Begin`/`End` pairs)" [6-3].
- "problems with the use of variables – L2 – not sure about declaring, initialising, and reading the variable again' [10-3].

Delphi resulted in some common syntax problems of its own: "L6 showed some confusion between control.property and property.control syntax, unsure of which to use. Had to be told that the computer needs to know which control first and then which property of that control" [30-6]. It could be that Socratic questioning was not being properly handled, in other words, the questions that the teacher asked were not directed enough, or that the knowledge required to answer the guided questioning was either unknown or not available (as it had not been flagged as important in the particular context). There is, thus, maybe, the need for explicit teaching of some of these concepts instead of allowing the construction of the concept work entirely from the use of multiple examples.

5.2.11 Limited success of demonstrating tasks (C11)

The middle phase of the Karplus cycle was occasionally replaced by a teacher demonstration (instead of a code review) in an attempt to find an approach that

would work best for learners with ADHD. In such cases, the teacher would demonstrate a skill, and it would then be up to the learners to implement the ideas in their own programs on their own PCs. The exercise would be appropriately chunked to accommodate for working memory limitations.

Demonstrations of code structures worked fine for simple tasks and for giving direction to learners: "got them round my PC and showed/added the `If` statement, `Begin`, `End`. ... Got them to implement the `If` statement in their own programs" [9-2]. The result was a class that "was enthusiastic about trying to run their code" [3-1]. Also, the simple process of creating a project folder and saving the project and code unit files worked well with this approach: "Demonstrated process, and then off to implement. Most coped fine" [20-1].

However, for more complex tasks, this did not work as well: in an exercise in creating procedures from existing code by cutting and pasting, "L1 and L3 seemed OK with the process. L2 looked blank. He struggled at his PC with deciding which bits needed to be cut" [11-2]. Also, when I started using demonstrations in the Delphi parts of the course, "I got the impression they were simply copying the code from my PC. They did not seem to understand much. I must try a different approach" [16-3].

Later I realised that learners were copying code from the teacher's PC because "I had not given them an example to follow or code to interpret ... – need to rethink this bit" [20-4]. This code example could then act as scaffolding as needed.

5.2.12 Hacking code as a sign of lack of planning know-how (C12)

In this context, "hacking" refers to programmers who simply launch into programming a solution without any planning whatsoever, thus relying on the compiler to pick out errors and the output to determine whether logic errors have been made – the program is then complete when it does what the programmer wanted, the code being hacked together with sections of code added as needed. When expanding a simple 'hello world' program, some learners "tried initially to write the code on paper, but

then rather tried to implement their ideas directly onto the PC" [3-3]. Also, after seeing a program that prints a line of 10 stars with a pause between each one run, learners described what the program did. They were then asked to "get around a table and write the code to produce it. Three sat around a table; L10 started working in DevPas. Once the other group had got some code down, they, too, went to PCs and tried to code" [6-1]. On another occasion, "I tried to get him (L6) to write down on paper what he had to do, but he preferred to hack it with my code as an example" [28-2].

Although the necessity of planning was discussed, "they wanted to just hack it. Discussed this with them and also discussed planning. They liked idea of hacking more" [10-3]. In discussions with the learners, we "talked about planning first – both felt it was essential, but still felt uncertain about how to proceed" [15-1]. Hacking, thus, was the result of learners not knowing how to plan on paper. Although planning was believed to be good, learners preferred to hack code.

5.2.13 Problems with scaffolding (C13)

Scaffolding was provided in various forms. Placing comments in code that learners had to replace with actual code sometimes worked and sometimes not. On one occasion, I noted: "I am convinced that providing commented program code (in apprenticed learning) helps them to learn faster – they get to experiment with working programs and learn quicker what crashes or succeeds in a program" [5-4]. In a later observation, however, they clearly had problems with this: "boys went off to their individual PCs to attempt the implementation – to add in the actual code in place of the program comments. Again, class struggled a bit to figure out what to do" [3-5].

Scaffolding as program steps written on bits of card, which then acted as manipulatives and could be moved into the correct program flow order, resulted in a "process [that] went fairly smoothly", although "when running the program, produced garbage, as they were stuck in a loop" [10-1]. When such scaffolding is available, though, learners do use it: "L5 and L6 used board/printed code as scaffolding where there is the basic structure of the code without the details ... Used this a lot" [24-1].

However, the copying of scaffolded code is not without its problems:

- “for L5, even with an example to copy, he made errors in the message box that he could not see” [25-3], even when the compiler noted errors in the statement.
- “even though the code example had used `If-Else`, L6 and L8 had used just three `If` statements – the code example did not guide” [31-4].

It went as far as “I think example code ... confused learners” [33-3]. An unintended result could have been that learners “found starting a program from scratch difficult. They were not always sure how to start” [15-1].

Was this a sign of ADHD behaviour where a teaching method works today but not tomorrow or a sign that the scaffolding was not provided in a manner that could guide the learners? Questions do remain, though: does copying code count as scaffolding, and does it lead to learning?

5.2.14 The place of direct instruction (C14)

Sometimes direct instruction has a role to play: “L1 was surprised at `ans:=A+B` with `A` and `B` declared as `string`, and after entering 3 and 4, got 34. ... older boys saw that it was simply ‘putting the numbers next to each other’. I explained that `A` and `B` were being seen as text, not numbers. Reminded them of Excel and column formatting” [4-2]. Here a simple explanation made it clear to all the learners, as well as introducing the terminology used and providing an opportunity to introduce integers. This was found to be particularly useful when a problem occurred: with a division-by-zero problem when finding an average, “L1 said – if the number is zero. I picked up on this, got them round my PC, and showed/added the `If` statement, `Begin`, and `End`. Debugged, and stepping through could show them how the program skipped sections when tests were not satisfied” [9-2].

There are some situations, such as local and global variables, where direct instruction of good programming technique becomes necessary: “Asked class about

variable declarations at bottom instead of top; why there? They were not able to answer, apart from saying it makes it easier to read, as variables are right above the program" [12-4] – showing why a variable has to be global would be simple enough, as it cannot be seen in parts of the program that require it, but showing that a global variable should be a local one without direct instruction in programs that would work in either case is more difficult.

5.2.15 Spontaneous experimentation as an indication of comfort with task at hand (C15)

The level of experimentation indicates the level of comfort that learners have with a topic. This can be seen from simple use of colour and variable names to program functionality. For example, learners "started experimenting with three numbers and division by zero" [4-5], where some liked "experimenting and seeing what happens" [5-2], playing around "especially with text colour" [9-2].

Such experimentation was seen more often in Delphi programs, where it is easier to change the look and feel of a program and the placement of, and captions on, buttons. For example, when using example code, it was "not just copied ... they were adding their own wording, and even purposely switching the functionality of Yes/No buttons" [37-2]. Also, "L1 noted that an edit box was being used for the answer and not a label. He used a label in his own code. He also used a feet-metres conversion instead of the °C-°F conversion in the project, as he needed it for himself" [38-4]. Also, here, experimentation involved going beyond what was asked in the task requirements, for example,

- taking "the program requirements further to include larger and smaller" [22-4] and
- "L1 did take his project further, to concatenating strings from buttons" [22-6].

5.2.16 Uncertainty about how to proceed (C16)

Insecurity about what code examples meant and how to move forward could be seen throughout the learning cycle, and this was often expressed by learners:

- “They got the idea of what is expected and what an average is, but did not know where to start” [33-3].
- “as before, had no clue where to start” [34-2].

In cutting and pasting code to create procedures, “L2 looked blank. He struggled at his PC with deciding which bits needed to be cut. ... He said he was not sure what the lines/code were meant to do and so was not sure how to find the blocks” [11-2]. This same learner found programming “complicated”, but was glad that “there was a friend he could talk to about what he was doing – a second opinion about stuff” [14-4].

A problem that learners found in Delphi was “with where the code goes – he (L6) had the code for the Go button in the radio button code” [24-3]. A learner who had been involved in both courses noted, however, that “people seem to be picking up the programming faster in Delphi” [31-3] as compared to the Pascal code of the first course.

A theme that often went along with this insecurity was the need for “more practice” [11-2] and [15-1]. This uncertainly was also linked to the preference for hacking noted in section 5.2.12. Further, it led to frustration as learners “got to a point and did not know how to move on” [15-3].

5.2.17 Multiple solutions to problems (C17)

Creative solutions were seen where there were greater levels of comfort and experimentation. For example, when “the `count` variable was not giving the correct reading ... both sorted out the problem: L1 set `count` to 1 at the start, L2 added one at the end” [10-2]. Also, “He (L1) has written a separate procedure to change the font to black on the radio buttons” [25-1], whereas others did not go this route. In the same vein, instead of leaving the displaying of the average until the end of the program, one learner “wanted to put it on a button request (not on the close button)

and realised that it may mean resetting `count` and `total`. We discussed displaying a running average in a separate label" [33-2].

In a sense, albeit at a more simple level, learners who "used their own names for buttons ... also using their own layout" [25-2] were creating a different solution to a problem.

5.2.18 Haphazard acquisition of programming conventions (C18)

Learners also had to learn the social conventions of programming as used by programmers in terms of variable naming standards and code layout and indentation. This was not formally taught. However, it was implicit in code examples and when in discussion with learners where there were problems with understanding what code did (such as `Begin-End` blocks lined up). Such conventions were taken up haphazardly.

Typical problems included

- the inconsistent use of "correct lowercase/uppercase naming conventions" [21-3] and
- the renaming of controls in Delphi: "L6 – some controls are being renamed (especially labels and buttons), but not others, and also not the form" [26-5].

However, where indentation was simple (only one level of indentation), it was often well handled. The importance of such conventions was noted by a learner when trying to trace a missing `End` statement: "While trying to trace it, he realised that `Begin-End` pairs not lined up made seeing what is what quite difficult" [35-2].

5.2.19 Pacing too slow (C19)

Sometimes the movement was too slow for some: "L1 did take his project further ... He also wanted to know when we will start putting things together – I got the idea he

is beginning to get frustrated and wants to move on" [22-6]. This was noted only once in the diaries.

5.2.20 Freedom to experiment (C20)

Learners felt that they no longer had to stick to the exact parameters of the requirements and felt free to experiment with layout and function – this was noted right from the beginning of the courses, but was more pronounced in the Delphi course. This is exemplified by the following comments: "L1 extended his program without concern. Many used their own names for buttons, that is, there exists the scope for own input. Some are also using their own form layout" [25-2].

5.2.21 Real-world systems to guide thinking (C21)

Thinking about how a program should operate must include thinking about how systems in the real world operate, and this knowledge is available to learners: in discussion about how to close/exit a program and the use of a close button, learners answered "yes, sometimes, but usually through File-Exit" [36-2]. Such knowledge can also be seen in the use of passwords: "L5 wanted to know whether you can lock people out after three tries with a password" [26-3].

5.2.22 Efficient use of the IDE as an indication of understanding of its structure (C22)

Learners showed comfort with the Delphi integrated development environment (IDE) with practice and used it to increase programming productivity. This can be seen in learners' use of the IDE in the following situations: instead of searching through all the code to find the appropriate code segment, as one learner did, "L6 returned to the form and double-clicked on the button to find the code – a more intelligent/reasoned/experienced response" [27-3]. The object inspector was also used "to find the names of components" [34-2].

5.3 Four emerging categories

In further open coding, these 22 concepts were grouped into four categories. It needs to be noted that concept C19 was not well represented in the data, and as it did not fit into one of the four categories discussed below neatly, it has been excluded from further analysis at this stage.

5.3.1 Using the Karplus learning cycle with refinements

This category is made up of the following concepts (as numbered in section 5.2): C3, C9, C10, C11, C13, and C14. At the centre of this category is the Karplus learning cycle (exploration, concept introduction, concept application), and as was pointed out above (in C9, section 5.2.9), the straightforward use of the Karplus cycle did not have the expected effect. Experimenting with the program did lead into the topic and give learners an idea of what could be achieved. Further, as was noted in C3 (section 5.2.3), once the idea had been grasped, it could be used in concept application.

However, the ability to apply the concepts in practical exercises at the end depended largely on how the middle step was handled – how did the presentation of the formal concept proceed, and was it understood/constructed by the learner? It was seen that the code syntax was often not self-explanatory (C10, section 5.2.10) and that other attempts such as demonstrations (C11, section 5.2.11) and scaffolding (C13, section 5.2.13) did not lead to sufficient understanding of the code structure being learnt. Simple presentation of the code and formal discussion of its use, thus, did not appear to lead to the required level of understanding to proceed comfortably to the next step in the cycle. Further, it appeared that there might well be a role for direct instruction in the cycle (C14, section 5.2.14). Programming conventions might also benefit from being taught directly (C18, section 5.2.18). The use of the Karplus learning cycle will be expanded on below in section 5.4 on the core category.

5.3.2 Thoughtful application of Socratic questioning

This category is made up of the following concepts: C1, C2, C4, and C5. Socratic questioning (C5, section 5.2.5) is at the centre of this category. This means that questions need to be based on knowledge that the learners already have or can get to via some guided (or Socratic) questioning (utilising the zone of proximal development).

However, such questions need also to be targeted at exactly what is required; they need to be specific and direct (C1, section 5.2.1). Further, questions should make clear the level of abstraction that is wanted in the answer, and this level of abstraction needs to be at a level with which learners can cope (C4, section 5.2.4). Careful and thoughtful use of questioning can also avoid situations where learners focus their attention on side issues rather than the task at hand (C2, section 5.2.2).

5.3.3 Learning time

This category is made up of concepts C6 and C7. Learners need time both to voice their understanding (C6, section 5.2.6) and to use McDermott's cycle with its predict-confront-resolve phases (C7, section 5.2.7). As noted in the section on instructional strategies for teaching learners with ADHD (section 3.7.1), such learners need time to organise an answer to a question, and the teacher needs to make this time available. Also, it takes time to handle a question asked by a learner, or solve a problem raised by a learner, via the more lengthy McDermott cycle (where simply giving an answer would go much quicker).

This category is linked to the category on questioning (section 5.3.2) in that the teacher needs to draw out a learner's understanding via questioning. Using McDermott's cycle also requires the teacher to ask the learners what they expect to happen, to help them (via questioning) to see and confront the actual, unexpected result, and then to use Socratic questioning to guide them to resolve the problem.

Thus, time must be provided for learners to express their understanding and expectations and to experiment with the compiler and grapple with compiler errors, and this takes courage on the part of the teacher, believing that it will lead to more effective constructions of the programming task by learners in the long run.

5.3.4 Encouragement of planning and experimentation

This category is made up of the following concepts: C8, C12, C15, C16, C17, C20, C21, and C22. There are two sides to this category: avoiding insecurity and building confidence. A focus on teaching and using planning can both lead to a reduction in hacking (C12, section 5.2.12) and provide some certainty about what the learner is supposed to do next (C16, section 5.2.16). Where there is insecurity, learners tend to sit back and listen (C8, section 5.2.8). Such non-participation invites opportunities for the ADHD learner to drift off task, which will only further complicate the learning process.

On the other hand, building confidence could be managed by creating an environment where there is freedom to experiment (C20, section 5.2.20), to encourage learners to be guided by what real-world systems require (C21, section 5.2.21), and to experiment with different solutions (C15, section 5.2.15 and C17, section 5.2.17). Confidence can also be built by ensuring an understanding of the structure of the IDE, leading to more efficient use of the IDE and comfort with the programming process (C22, section 5.2.22).

Thus, comfort with the programming process from planning to execution indicates the level of knowledge of, and comfort with, the programming process and its composite parts. This includes considering how real systems work and attempting to incorporate these concepts into a solution.

5.4 The core category: using the Karplus learning cycle with refinements

The Karplus learning cycle category would clearly be the core category, as it was the focus of the action research project in the first place. Also, the other three categories identified from the initial concepts found all fit around this core category.

- The use of Socratic questioning can be seen in all phases of the Karplus learning cycle: using questions that guide learners in the exploration phase, ensuring that they get maximum exposure to the concepts and skills that are being demonstrated in the new program; guided questioning would clearly be used in the concept introduction phase, as it builds on previous knowledge and exploration in creating new knowledge; and finally, it would be used in concept application, as learners are referred back to the previous two phases as the skills are applied in new situations.
- Learners would need time throughout the learning cycle to properly explore the abilities of the new program, to learn new concepts, and to find uses in new applications. This time would be both to express what they are learning and to use McDermott's predict-confront-resolve cycle to construct new understandings.
- Although there would not be much need for planning in the exploration phase of the learning cycle, there certainly would be a need for experimentation. The real use of planning and experimentation in the building of confidence would be seen in the concept application phase of the cycle.

5.4.1 The first and third phases (exploration and concept application)

As noted in section 5.3 above, the Karplus cycle's first and last phases worked effectively in the teaching of introductory programming. In particular, the use of a running program provided an engaging task that was needed to gain the attention of the learner with ADHD, it acted as an example of what would be expected of learners, and it could also be a form of cueing and setting the scene for what was to come next – all techniques that help learners with ADHD learn. Similarly, during the third phase when learners were practicing the newly learnt skill, the teacher had an opportunity to move around the class looking for signs that learners were moving off task and to use appropriate ADHD behaviour techniques to bring the learner back on task.

5.4.2 The second phase (concept introduction)

It was, however, in the middle phase where learner understanding fell short of what was required. Attempts at providing the code and working through it constructively (individually, in pairs, and/or as a group) to build a mental model of the programming structure were not successful: learners' attention tended to drift, and they waited for someone else in the class to offer answers or questions, often did not see the significance of the program code to the solving of the problem or the running of the program, and were even confused by the code. This effect was more obvious in the case of Delphi where there is a lot more extraneous code to get a program to run. In Pascal, however, the code is more straightforward, albeit less like a traditional Windows application when run from a user point of view. Attempts to use the code, then, as the formal introduction to the programming concept did not work as intended and were not the guide to the next phase of the learning cycle.

It was noted in the observations that concepts were grasped via interacting with examples, although, at times, some learners saw the significance of the example and some did not. The question, then, is how to provide sufficient examples to allow for understanding by the whole group within the Karplus cycle. Direct instruction of programming structures could provide a focus on the important parts of the coding structure sufficiently to allow learners to understand their significance. There is the possibility that direct instruction could also be an opportunity to provide learners with another example of the code structure under discussion. Such instruction would, thus, add a further example to those already provided in exploration and in demonstrations provided by the teacher, on condition that the teacher carefully chooses the examples in the different phases to allow learners to see the code operating in different situations.

Teacher demonstrations on a computer on how to type up the programming structure provided an apprenticeship in the concept required. It allowed the teacher to demonstrate the thinking process behind the use and structure of the code, as well as the handling of compiler error messages. When such demonstrations were attempted, it was observed that when the code was fairly simple (a line or two of code), the learners were able to reproduce the code without difficultly. However, when more complex structures were demonstrated, learners resorted to simply copying the code from the teacher's computer, showing little understanding of the

programming structure itself. This lack of understanding was especially obvious when code was copied incorrectly from the teacher's demonstration computer. It could be that demonstrations by themselves may not be enough where there are limited working memory problems and that scaffolding could be provided with some example code that sets out the basic structure of the required code.

5.5 A possible expansion of the Karplus learning cycle

Now that the grounded theory coding process has allowed for evaluation and specifying learning in the first action research cycle, a new phase of diagnosing can begin and action planned around the diagnosis.

As neither demonstrations nor code walk-throughs were effective by themselves, it is proposed that a step be added to the middle phase of the Karplus learning cycle. The two diagrams in Figure 6 are two possible alternatives, representing a more guided practical phase and a more theoretical phase in the middle part of the cycle. The exploration phase would be retained in the form that has been used so far – experimenting with a running program to explore what the computer can achieve using a technique that has not been used yet. From here, there are then two options:

- Move into a demonstration of using the new technique (Figure 6A). This would be chunked as necessary, and the end product would be a working program that uses the new technique. The program that is created in this phase could be the same one that learners experimented with in the exploration phase or a completely new program.
- Move into a formal, direct instruction phase that highlights the important aspects of the code structure (Figure 6B). Where this type of technique/code structure could be used in other problems requiring a programming solution could also be covered in this part of the cycle. In this phase, learners could possibly use cloze notes (see section 3.7.1) that would be completed as the instruction progresses. Further, this would be an opportunity to focus on the programming conventions that would be associated with the programming structure being learnt.

From here, the learners would progress to the third proposed phase. If, from the exploration, learners had moved to a demonstration phase, they would now learn the formal terminology and structure of the concept through direct instruction (Figure 6A). On the other hand, if learners had first learnt the concept via direct instruction, they would now move to a demonstration phase (Figure 6B).

The last phase would be the same for the two different approaches and similar to the last phase of the original Karplus learning cycle: the concept application phase, where they would practise what had been learnt in the exploration, demonstration, and instruction phases (though not necessarily in this order).

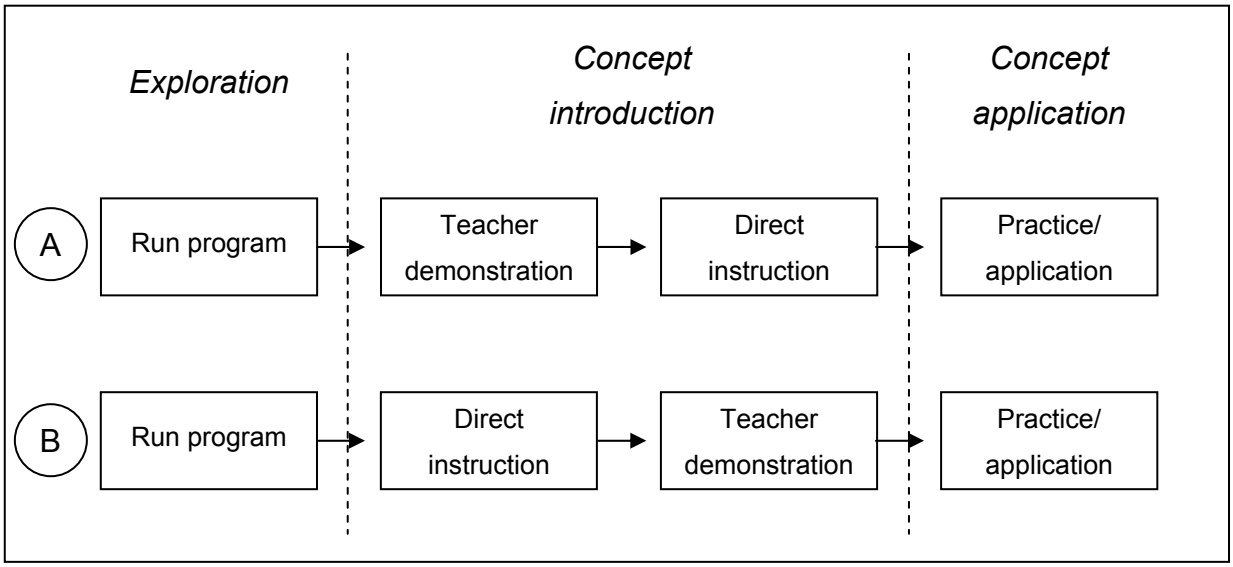

Figure 6 Possible expansions of the Karplus learning cycle

Further research will be necessary to determine which works better or which should be used in which particular educational circumstances. However, both approaches allow for the varying of lesson presentation strategies with alternating periods of sitting still and active participation needed by learners with ADHD. Further, the teacher demonstrations, which would include opportunities for learners to implement the code as the demonstration progressed, would allow the teacher to chunk the material to the appropriate level for the learners and teach the necessary programming organisational skills (such as creating directories to keep all the program files together).

It is possible that a multiple approach strategy would serve best and would compensate for when examples are understood by some learners and not by others, thus allowing a multiple entry approach to the programming construct under discussion. This approach would be in keeping with learning strategies used with learners with ADHD, especially remembering that techniques that worked one day are unlikely to work all the time. An example of this approach (based on Figure 6A, although it could as easily be used in the approach in Figure 6B) is developed in Figure 7. The teacher would not be tied to using only one of the approaches in each phase, and it may well turn out that more than one will be required to ensure understanding of the programming construct that is being learnt. The final picture of a workable approach could be a string of overlapping events run in series rather than a few discrete phases.

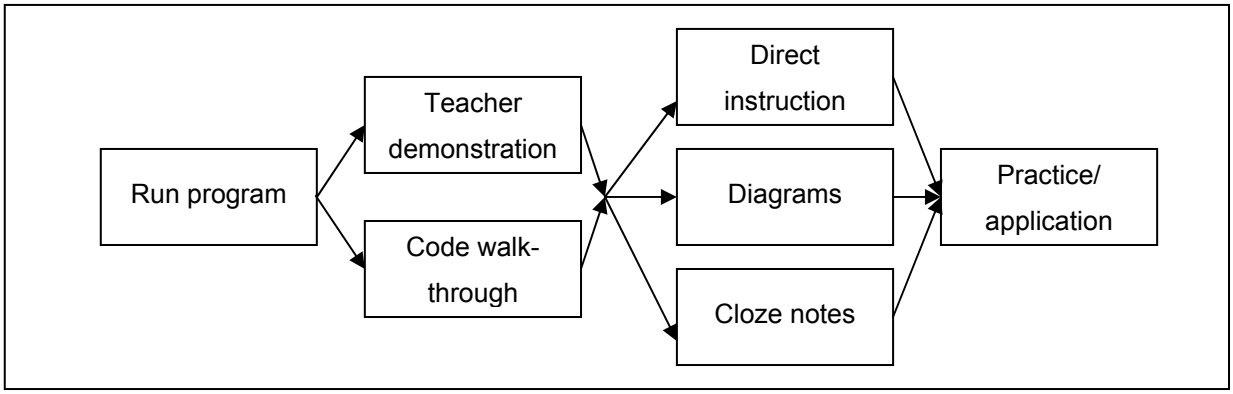

Figure 7 Using multiple approaches in the learning cycle

5.6 Conclusion

Grounded theory methods were used to complete the last two phases of the first action research cycle: evaluation and specifying learning. Through open coding, 22 concepts relating to learning and teaching of introductory programming were identified. These were grouped into four main categories. I found the Karplus learning cycle to be the core category around which the other three categories could be built and integrated. This finding was not surprising, as the Karplus learning cycle was the focus of the research in the first place, and it would be expected that grounded theory methods should highlight this as the core category.

The diagnosis phase of the second action research cycle led to apparent inadequacies in the concept introduction phase of the Karplus learning cycle. Action planning opened up the possibility of extending the concept introduction phase of the learning cycle to include practical and theoretical aspects of the programming concept under discussion.

This has implications for the use of constructivist learning cycles when teaching learners with ADHD introductory computer programming. It is to these implications and conclusions that I now turn.

Chapter 6 Conclusions

6.1 Introduction

Within the constructivist paradigm, teaching is not just about the passing on of information from the teacher to the learner, but rather the designing of learning environments that will allow learners to construct their own mental models, merging their horizons with that of the teacher (to use the terminology of Feldman (1994)). The goal of teacher research would, thus, be the transformation of teaching practice (Doerr & Tinto, 2000), leading to improved practice and a better understanding of the educational environments that are being created (Feldman, 1994).

This study was a result of an interpretive research project using action research, where grounded theory was used to analyse the data gathered during the research. Interpretive research has been found to be useful in understanding human thought and action and is, thus, suited to the educational nature of the current research (Klein & Myers, 1999). The action research approach, research that comes from "the perspective of practice" (Lampert in Ball, 2000, p. 366), guided the research and provided the structure, ensuring a systematic enquiry. Using the grounded action research methodology proposed by Baskerville and Pries-Heje (1999) led to the refining of the practice of action research. It is from this process that I now draw conclusions.

In this conclusion, I will summarise the findings of the study and evaluate it in terms of a framework suggested by Klein and Myers (1999). Conclusions will then be drawn and contributions summarised. Suggestions for future work will also be presented.

6.2 Summary of findings

Four specific questions were asked at the beginning of the study and have been addressed in the chapters above. The findings can be briefly summarised as follows:

1. What is constructivism, and how can it be used to support the teaching and learning of programming?
2. What is the Karplus learning cycle, and how can it be used to support the teaching and learning of computer programming?

   Building a mental model of the programming task can be approached from a constructivist point of view, where learners grapple with the practical applications of the programming concept before learning the terminology and formal form of the particular programming structure. The Karplus learning cycle is one constructivist approach to structuring educational activities, which is one approach to teaching and learning. Further, there are constructivist techniques that can be used to teach introductory programming, and these were employed in the teaching of the courses on which this study was based.

3. What is ADHD, and what are the strategies that can be used to promote learning by children with ADHD?

   Learners with ADHD face challenges to learning that can be ameliorated by using teaching and learning strategies that have been shown to be successful in such educational settings. However, there are no foolproof methodologies, and no one intervention is going to increase learning effectiveness all the time.

4. Is the Karplus constructivist learning cycle successful in the teaching of computer programming to children with ADHD?

   Constructivist and ADHD teaching techniques were put together using the Karplus learning cycle to structure teaching and learning. In the courses that followed this approach, it was found that though the exploration and concept application phases of the Karplus cycle worked as expected, the middle phase, concept introduction, needed to be expanded to be more effective. Concept introduction could not be understood as a single phase, but subphases may have to be added to ensure an acceptable introduction to the programming concept under discussion.

6.3 Evaluation of the study

I will briefly evaluate this research using the principles for evaluating interpretive field studies (Klein & Myers, 1999). Although these principles were intended to be used to evaluate field studies in information systems, I believe that they provide a good starting point from which to evaluate an action research project in computer programming education.

The principles of evaluation are given below with a brief discussion in terms of the current research.

6.3.1 The hermeneutic circle

This is the fundamental principle on which much of the rest of the principles rests. Its premise is that interpretation moves from a preliminary understanding of the parts and their interrelationships to the whole and then from an understanding of the bigger picture back to an improved understanding of the parts again. It has also been argued that the concepts 'parts' and 'whole' be given a "broad and liberal interpretation" (Klein & Myers, 1999, p. 71).

In this study, the parts can be seen as the phases of the Karplus learning cycle, as well as the other three categories that were identified in this study. How these parts interrelate and together make up the whole of the learning process and how the learning process can be subdivided into these parts make up this hermeneutic circle. Further, the various concepts that make up the Karplus cycle category itself could be seen as the parts of the whole category and that the category (the core category in this study) could be understood in terms of the concepts that make up the core category. It is, then, in the movement between these various levels of interpretation that one gains a better understanding of the learning process of teaching ADHD learners introductory programming.

6.3.2 Contextualisation

This principle is focused on the social and historical background of the research setting. Thus, there should be reflection on, rather than avoidance of, the differences in understanding between participants and interpreter in the study, understandings of programming, and understandings of learning in this particular study. Further, there must be a recognition of the historical context and how the current situation emerged.

Although there is a focus on the learner with ADHD in the classroom context, and an acceptance of the fact that the learner, together with the teacher, is involved in the creation of the learning environment, there could be more critical focus in this area. For example, this study could be extended by considering why so few learners actually took the extramural activity compared to the many who showed interest initially. There could be also be an exploration of the historical context and whether there is the perception that computer programming is too difficult for learners with ADHD to attempt.

6.3.3 Interaction between researchers and subjects

The requirement of this principle is that the researcher critically evaluates to what extent the data was a social product of the interaction between the researcher and the participants. The suggestion is, then, that fundamentally the data collected and the knowledge generated exist only in the relationship and interaction between the researcher and the social context of the participants. Further, participants "are interpreters as they alter their horizons by the appropriation of concepts used by ... researchers, ... and they are analysts in so far as their actions are altered by their changed horizons" (Klein & Myers, 1999, p. 74).

As this was an action research study (rather than a field study) where the researcher intervened in the learning process and noted carefully the response to the intervention, it is believed that there was sufficient interaction between the researcher/teacher and the participants/learners. This can also be seen in the

merging of horizons of both teacher and learner that Feldman (1994) sought and that was a guiding principle in this study.

6.3.4 Abstraction and generalisation

Although it is true that interpretive research values unique environments, this principle requires that the research still be linked to theoretical and general principles in the field of study. This requires relating the particulars of the current situation to abstract categories that apply to many different situations. Thus, a major point of this principle is that theory plays an important role in such interpretive research, where it can be a "sensitizing device" (Klein & Myers, 1999, p. 75) as the researcher interprets the environment.

Any classroom situation is a unique instance, and what happened has been related to general theoretical principles that relate to both constructivism and principles for teaching learners with ADHD. Further, such studies are not dependent on the representativeness of the cases, but on the reasoning of the argument. This has also been dealt with in section 4.11 on the validity and reliability of the research.

6.3.5 Dialogical reasoning

The dialogical reasoning principle requires researchers to be sensitive to their own preconceptions and prejudices, especially as these relate to the theoretical underpinnings that guided the original research design. This background needs to be made clear so that any contradictions between it and the actual data can be dealt with responsibly. This is important, as the intellectual basis, as well as any preheld ideas about the participants and their situation, will colour what is seen and recorded in the process of the research. Again, this is not so that such prejudices can be avoided, but so that they can be made clear from the outset, realising that they are part and parcel of the research process.

Although the theoretical and philosophical underpinnings of the research were covered in Chapter 4, this principle was probably not included to the extent that it

could be. This study could, thus, be enhanced by examining how preconceptions guiding the research may have affected the data collected and its interpretation.

6.3.6 Multiple interpretations

The requirement of this principle is that the researcher has to examine the various social and historical influences impinging on the study environment and seek multiple viewpoints and interpretations and the reasons for them. This means that the researcher needs to be sensitive to the different interpretations that participants may have of the context and the possibility of contradictions between them (should they exist). This principle is valuable, as it encourages researchers to probe beneath the surface and not to take events at face value.

The existence of multiple views on what is happening can be seen in interpretations that lie in the nature of ADHD learners, in the process of constructing knowledge, and in the relationship of the two where learners with ADHD are expected to learn via constructivist methods. However, it needs to be remembered that all interpretations are carried out within the teacher's understanding of both these bodies of knowledge, and there is the possibility of another interpretation lying here that has not been carefully examined.

6.3.7 Suspicion

This principle requires a researcher to "'read' the social world behind the words of the actors, a social world that is characterized by power structures, vested interests, and limited resources to meet the goals of various actors who construct and enact this social world" (Klein & Myers, 1999, p. 78). This principle has not been examined at all in this study. It seeks the real story behind what is being said by learners in the classroom, looking for possible false realities and vested interests that are simply accepted by the learners and the teacher.

6.3.8 Summary

These seven principles help provide a balanced view of the research that was undertaken. There are certainly areas that could have been examined in more detail, such as learners' views of computer programming and how the teacher's preconceptions of teaching and learning, computer programming, and ADHD influenced the presentation of constructivist approaches to teaching. However, these were beyond the scope of the present study, and it is further believed that they do not fundamentally affect the results that were obtained. The focus of the study was on the Karplus learning cycle and its use to structure learning environments, and it is believed that what happened in the classroom regarding the learning cycle and its effects has been honestly and fairly reflected.

6.4 Conclusions

The findings of this research show that constructivism (in the form of the Karplus learning cycle) can be used to teach learners with ADHD introductory programming. However, the learning cycle may need to be refined slightly.

Using the techniques suggested for teaching learners with ADHD was useful in organising the constructivist teaching approaches. Knowledge of these techniques allowed the teacher to be aware of when attentional problems could be a problem and to plan to avoid them. These techniques also prepared the teacher for behaviours associated with learners with ADHD and gave the teacher tools to more effectively assist in the learning.

Together with the techniques for better helping learners with ADHD learn, constructivist approaches to the task of teaching were also effectively used in getting away from pure teacher-led information transmission. Although the use of constructivist approaches was found to require some courage on the part of the teacher, they also needed time to be used effectively.

Knowledge of constructivist and ADHD learning approaches comes together in the use of the Karplus learning cycle to teach learners with ADHD introductory programming. It has been concluded that this learning cycle can be used to facilitate the learning of introductory programming by learners with ADHD. The first phase, concept introduction, worked well in the form of a running program that caught the attention of the learners. The last phase, concept application, provided learners with opportunities to practise their newly learnt skills.

However, the middle phase, concept introduction, will need to be understood more broadly. It was found that this phase of the cycle might need to be understood as two separate subphases that cover both practical and theoretical aspects of understanding. It is also possible that the concept introduction phase could be made up of several different activities and that these should be used as needed to ensure effective learning in the context of learners with ADHD.

The findings, based as they are on a small number of participants in an action research study, will not be generalisable to a population, but can contribute to general constructivist education theory (Baskerville & Pries-Heje, 1999; Corbin & Strauss, 1990). They show that constructivist methods can be used with learners with ADHD, although concerns about whether these learners will be able to construct the necessary knowledge still need to be taken seriously. This can be seen in the need to extend the concept introduction phase of the learning cycle, particularly in the introduction of more formal, direct instruction into the cycle.

Practically, this study does show that teachers can move away from the language-constructs view of most computer programming/language textbooks and use an approach that is based more on projects, introducing concepts as they are needed to complete the set problem. Of course, the sequencing of the problems will still have to be carefully structured to ensure that learners are not overwhelmed by the volume of new knowledge that has to be constructed.

It does need to be noted that the contributions offered here are based on a particular approach to the Karplus cycle and with a particular, and small, group of learners who were learning introductory programming within a Delphi (and Pascal) environment.

Although it is conceivable that similar results would be obtained with other similar special needs learners using another programming language (such as Java, for instance), only attempts in such educational environments will tell.

Thus, the Karplus learning cycle can be used, with care, to facilitate the learning of introductory computer programming by children with ADHD.

6.5 Future studies

There are possibilities for future studies inherent in this work. One that was noted in the evaluation above (in section 6.3.2) deals with perceptions of computer programming: do learners with ADHD view computer programming as too difficult for them to attempt? And if so, where does this perception come from?

This study could also be continued into the second action research cycle, where the proposed extension of the Karplus learning cycle is implemented and the responses and results noted and analysed. This would, no doubt, lead to other findings that would, in turn, lead to new learning and action planning and another action research cycle. Repeated cycles could then lead to the refinement of the use of the Karplus learning cycle for learners with ADHD learning introductory computer programming.

The question as to whether there is a general model that can be used to teach learners with ADHD introductory programming could also be explored further. Such a model could relate to the relative importance of individual or group work and the place of pure discovery and direct teaching in the construction of a viable model for building programs.

6.6 Conclusion

One of the advantages of qualitative studies such as this one is that findings may take one in an unexpected direction (Hazzan et al., 2006). Where the simple verification of the value of the Karplus learning cycle may have been expected, the broadening of the breadth of the cycle may not have been foreseen and would need

to be confirmed in further studies. Further, such findings are likely to benefit other learners (apart from those with ADHD) as well, thus broadening the scope of the findings beyond the limited confines of the current study.

References

Abell, S.K. (2005). University science teachers as researchers: blurring the scholarship boundaries. *Research in Science Education, 35*, 281-298.

Alban-Metcalfe, J. & Alban-Metcalfe, J. (2001). *Managing attention deficit/hyperactivity disorder in the inclusive classroom. Practical strategies for teachers.* London: David Fulton.

Allen, K. (2005). Online learning: constructivism and conversation as an approach to learning. *Innovations in Education and Teaching International, 42*(3), 247-256.

Almstrum, V.L., Guzdial, M., Hazzan, O. & Petre, M. (2005). Challenges to computer science education research. *ACM SIGCSE Bulletin, 37*(1), 191-192.

American Psychiatric Association. (1994). *Diagnostic and statistical manual of mental disorders* (4th ed.). Washington: Author.

Andrew, A.M. (2004). Questions about constructivism. *Kybernetes, 33*(9/10), 1392-1395.

Armstrong, T. (1996). ADD: does it really exist? *Phi Delta Kappan, 77*(6), 424-429.

Astrachan, O. (1998). Concrete teaching: hooks and props as instructional technology. *ACM SIGCSE Bulletin, Proceedings of the 6th Annual Conference on the Teaching of Computing and the 3rd Annual Conference on Integrating Technology into Computer Science Education (ITiCSE'98), 30*(3), 21-24.

Attention deficit disorder: old questions, new answers. (2006, February). *Harvard Mental Health Newsletter*, 3-6.

Avison, D., Lau, F., Myers, M. & Nielsen, P.A. (1999). Action research. *Communications of the ACM, 42*(1), 94-97.

Ball, D.L. (2000). Working on the inside: using one's own practice as a site for studying teaching and learning. In A.E. Kelly & R.A. Lesh, (Eds.), *Handbook of research design in mathematics and science education* (pp. 365-402). Mahweh, New Jersey: Lawrence Erlbaum.

Baskerville, R. & Pries-Heje, J. (1999). Grounded action research: a method for understanding IT in practice. *Accounting Management and Information Technologies, 9*, 1-23.

Baskerville, R.L. & Wood-Harper, A.T. (1996). A critical perspective on action research as a method for information systems research. *Journal of Information Technology, 11*, 235-246.

Ben-Ari, M. (1998). Constructivism in computer science education. *ACM SIGCSE Bulletin, Proceedings of the 29th SIGCSE Technical Symposium on Computer Science Education SIGCSE'98, 30*(1), 257-261.

Ben-Ari, M., Berglund, A., Booth, S. & Holmboe, C. (2004). What do we mean by theoretically sound research in computer science education? *SIGCSE Bulletin, Proceedings of the 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, ITiCSE'04, 36*(3), 230-231.

Bester, H. (2000). *Help, my child is causing chaos! Hyperactive? Or creative? A practical guide.* Cape Town: Human and Rousseau.

Blum, F.H. (1955). Action research − a scientific approach? *Philosophy of Science, 22*(1), 1-7.

Bodner, G.M. (1986). Constructivism: a theory of knowledge. *Journal of Chemical Education, 63*(10), 873-878.

Brand, S., Dunn, R. & Greb, F. (2002). Learning styles of students with attention deficit hyperactivity disorder: who are they and how can we teach them? *The Clearing House, 75*, 268-273.

Burgess, J. (2003). Pay attention please! Strategies to reduce the learning difficulties experienced by children with attention and concentration problems. *Australian Journal of Learning Disabilities, 8*(3), 8-13.

Castellanos, F.X., Sonuga-Barke, E.J.S., Milham, M.P. & Tannock, R. (2006). Characterising cognition in ADHD: beyond executive dysfunction. *TRENDS in Cognitive Science, 10*(3), 117-123.

Cey, T. (2001). Moving towards constructivist classrooms. Retrieved October 13, 2006 from http://www.usask.ca/education/coursework/802papers/ceyt/ceyt.htm

Chua, W.F. (1986). Radical developments in accounting thought. *The Accounting Review, 61*(4), 601-632.

Clancy, M., Stasko, J., Guzdial, M., Fincher, S. & Dale, N. (2001). Models and areas for CS education research. *Computer Science Education, 11*(4), 323-341.

Clear, T. (2001). Research paradigms and the nature and meaning of truth. *inroads – The SIGCSE Bulletin, 33*(2), 9-10.

Clement, J.M. (2004). A call for action (research): applying science education research to computer science instruction. *Computer Science Education, 14*(4), 343-364.

Conducting content analysis. (2006). Retrieved April 23, 2006 from http://writing.colostate.edu/guides/research/content

Confrey, J. (1990). What constructivism implies for teaching. *Journal for Research in Mathematics Education*, Monograph 4, 107-122.

Corbin, J. & Strauss, A. (1990). Grounded theory research: procedures, canons, and evaluative criteria. *Qualitative Sociology, 13*(1), 3-21.

Daniels, H. (2001). *Vygotsky and pedagogy*. London: RoutledgeFalmer.

Davis, G.B., Lee, A.S., Nickles, K.R., Chatterjee, S., Hartung, R. & Wu, Y. (1992). SOS. Diagnosis of an information system failure. A framework and interpretive process. *Information & Management, 23*, 293-318.

De Villiers, M.R. (2005). Three approaches as pillars for interpretive information systems research: development research, action research and grounded theory. *ACM International Conference Proceedings Series Vol 150, Proceedings of the 2005 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries*, 142-151.

Doerr, H.M. & Tinto, P.P. (2000). Paradigms for teacher-centered, classroom-based research. In A.E. Kelly & R.A. Lesh, (Eds.), *Handbook of research design in mathematics and science education* (pp. 403-427). Mahweh, New Jersey: Lawrence Erlbaum.

Doolin, B. (1998). Information technology as disciplinary technology: being critical in interpretive research on information systems. *Journal of Information Technology, 13*, 301-311.

Driver, R., Asoko, H., Leach, J., Mortimer, E. & Scott, P. (1994). Constructing scientific knowledge in the classroom. *Educational Researcher, 23*(7), 5-12.

DuPaul, G.J. & White, G.P. (2006). ADHD: behavioral, educational, and medication interventions. *The Education Digest, 71*(7), 57-60.

Duveskog, M., Sutinen, E., Tedre, M. & Vesisenaho, M. (2003). In search of contextual teaching of programming in a Tanzanian secondary school. *Proceedings of the 33rd ASEE/IEEE Frontiers in Education Conference*, Session F3B, 9-14.

Feldman, A. (1994). Erzberger's dilemma: validity in action research and science teachers' need to know. *Science Education, 78*(1), 83-101.

Feldman, A. & Capobianco, B. (2000). Action research in science education. *ERIC Digest ED463944*. Retrieved April 23, 2006 from http://eric.ed.gov

Feldman, A. & Minstrell, J. (2000). Action research as a research methodology for the study of the teaching and learning of science. In A.E. Kelly & R.A. Lesh, (Eds.), *Handbook of research design in mathematics and science education* (pp. 429-455). Mahweh, New Jersey: Lawrence Erlbaum.

Flavell, J.H., Miller, P.H. & Miller, S.A. (1993). *Cognitive development.* New Jersey: Prentice-Hall.

Fleury, A.E. (1991). Parameter passing: the rules the students construct. *ACM SIGCSE Bulletin, Proceedings of the 22nd SIGCSE Technical Symposium on Computer Science Education, 23*(1), 283-286.

Gay, L.R. & Airasian, P. (2000). *Educational research: competencies for analysis and application* (6th ed.). New Jersey: Prentice-Hall.

Gigout-Hues, L. (2006). ADHD: a crash-free course. *www.TeachingK-8.com*, April, 54-55.

Gijbels, D., Van de Watering, G., Dochy, F. & Van den Bossche, P. (2006). New learning environments and constructivism: the students' perspective. *Instructional Science, 43*, 213-226.

Gonzalez, G. (2004). Constructivism in an introduction to programming course. *Journal of Computing Sciences in Colleges, 19*(4), 299-305.

Hallowell, E.M. & Ratey, J.J. (1999). 50 tips on the classroom management of attention deficit disorder. *The Southern African Association for Learning and Educational Difficulties, 18*(1), 5-8.

Harris, K.R. & Alexander, P.A. (1998). Integrated, constructivist education: challenge and reality. *Educational Psychology Review, 19*(2), 115-127.

Harris, K.R. & Graham, S. (1994). Constructivism: principles, paradigms, and integration. *The Journal of Special Education, 28*(3), 233-247.

Harris, K.R. & Graham, S. (1996a). Constructivism and students with special needs: issues in the classroom. *Learning Disabilities Practice, 11*, 134-137.

Harris, K.R. & Graham, S. (1996b). Memo to constructivists: skills count, too. *Educational Leadership, 53*(5), 26-29.

Hazzan, O., Dubinsky, Y., Eidelman, L., Sakhnini, V. & Teif, M. (2006). Qualitative research in computer science education. *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education, SIGCSE'06*, 408-412.

Hendry, G.D. (1996). Constructivism and educational practice. *Australian Journal of Education, 40*(1), 19-45.

Hoepfl, M.C. (1997). Choosing qualitative research: a primer for technology education researchers. *Journal of Technology Education*, *9*(1). Retrieved April 23, 2006 from http://scholar.lib.vt.edu/ejournals/JTE/v9n1/hoepfl.html

Hughes, I. (1996). *How to keep a research diary*. Retrieved February 13, 2006 from http://www.scu.edu.au/schools/gcm/ar/arr/arow/rdiary.html

Hult, M. & Lennung, S. (1980). Towards a definition of action research: a note and bibliography. *Journal of Management Studies, 17*(2), 241-250.

Jakovljevic, M., Ankiewicz, P. & De Swardt, E. (2003). Action research in an information systems design context: exploring a variety of instructional strategies and techniques to enhance technological problem solving. *Proceedings of the World Congress on Action Learning, Action Research, Process Management and Participatory Action Research*. Retrieved October 3, 2005 from http://www.education.up.ac.za/alarpm/PRPpdf/Jakovljevic, %20Ankiewicz&%20Swardt.pdf

Jick, T.D. (1979). Mixing qualitative and quantitative methods: triangulation in action. *Administrative Science Quarterly, 24*, 602-611.

Jonassen, D.H. (1991). Objectivism versus constructivism: do we need a new philosophical paradigm? *Educational Technology Research and Development, 39*(3), 5-14.

Karagiorgi, Y. & Symeou, L. (2005). Translating constructivism into instructional design: potential and limitations. *Educational Technology & Society, 8*(1), 17-27.

Karplus, R. (1980). Teaching for the development of reasoning. *Research in Science Education, 10*, 1-9.

Kavale, K.A. & Forness, S.R. (1992). History, definition, and diagnosis. In N.N. Singh & I.L. Beale, (Eds.), *Learning disabilities. Nature, theory, and treatment* (pp. 3-43). New York: Springer-Verlag.

Klein, H.K. & Myers, M.D. (1999). A set of principles for conducting and evaluating interpretive field studies in information systems. *MIS Quarterly, 23*(1), 67-93.

Kock, N., Avison, D., Baskerville, R., Myers, M. & Wood-Harper, T. (1999). IS action research: can we serve two masters? *Proceedings of the 20th International Conference on Information Systems*, 582-585.

Kock, N.F., McQueen, R.J. & Scott, J.L. (2006). *Can action research be made more rigorous in a positivist sense? The contributions of an interactive approach*. Retrieved February 2, 2006 from http://www.scu.edu.au/schools/gcm/ar/arr/arow/kms.html

Kölling, M. & Barnes, D.J. (2004). Enhancing apprentice-based learning of Java. *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education, SIGCSE'04,* 286-290.

Loyens, S.M.M., Rikers, R.M.J.P. & Schmidt, H.G. (2006). Students' conceptions of constructivist learning: a comparison between a traditional and a problem-based learning curriculum. *Advances in Health Sciences Education, 11*, 365-379.

Lui, A.K., Kwan, R., Poon, M. & Cheung, Y.H.Y. (2004). Saving weak programming students: applying constructivism in a first programming course. *inroads – The SIGCSE Bulletin, 36*(2), 72-76.

Martin, P.Y. & Turner, B.A. (1986). Grounded theory and organisational research. *Journal of Applied Behavioral Science, 22*(2), 141-157.

Mayes, S.D., Calhoun, S.L. & Crowell, E.W. (2000). Learning disabilities and ADHD: overlapping spectrum disorders. *Journal of Learning Disabilities, 33*(5), 417-424.

McDermott, L.C. (1991). Milikan Lecture 1990: What we teach and what is learned – closing the gap. *American Journal of Physics, 59*(4), 301-315.

McDougall, A. & Boyle, M. (2004). Student strategies for learning computer programming: implications for pedagogy in informatics. *Education and Information Technologies, 9*(2), 109-116.

McEwan, E.K. (1998). *The principal's guide to attention deficit hyperactivity disorder.* Thousand Oaks: Corwin Press.

McKnight, C., Magid, A., Murphy, T.J. & McKnight, M. (2000). *Mathematics education research: a guide for the research mathematician*. Providence, Rhode Island: American Mathematical Society.

Mouton, J. (2001). *How to succeed in your master's and doctoral studies. A South African guide and resource book.* Pretoria: Van Schaik.

Myers, M.D. (1994). A disaster for everyone to see: an interpretive analysis of a failed IS project. *Accounting, Management and Information Technologies, 4*(4), 185-201.

Myers, M.D. (1997). Qualitative research in information systems. *MIS Quarterly, 21*(2), 241-242. Updated version retrieved February 13, 2007 from http://www.qual.auckland.ac.nz

Naparstek, N. (2002). *Successful educators. A practical guide for understanding children's learning and mental health issues*. Westport, Connecticut: Bergin & Garvey.

Newman, J.M. (2000). Action research: a brief overview. *Forum: Qualitative Social Research*. Retrieved March 10, 2006 from http://qualitative-research.net/fqs

Orlikowski, W.J. (1993). CASE tools as organizational change: investigating incremental and radical changes in systems development. *MIS Quarterly, 17*(3), 309-340.

Padak, N. & Padak, G. (2006). *Research to practice: guidelines for planning action research projects*. Retrieved February 13, 2006 from http://archon.educ.kent.edu/Oasis/Pubs/0200-08.htm

Pandit, N.R. (1996). The creation of theory: a recent application of the grounded theory method. *The Qualitative Report, 2*(4). Retrieved February 18, 2007 from http://www.nova.edu/ssss/QR/QR2-4/pandit.html

Pears, A., Daniels, M. & Berglund, A. (2002). Describing computer science education research: an academic process view. *Conference on Simulation and Multimedia in Engineering Education (ICSEE)*, 99-104. Retrieved October 3, 2005 from http://user.it.uu.se/~anderssb/publications/The_model.pdf

Perkins, D.N. (1991). What constructivism demands of the learner. *Educational Technology, 31*(9), 19-21.

Perkins, D.N. (1999). The many faces of constructivism. *Educational Leadership, 57*(3), 6-11.

Phillips, D.C. (1995). The good, the bad, and the ugly: the many faces of constructivism. *Educational Researcher, 24*(7), 5-12.

Powers, K.D. (2004). Teaching computer architecture in introductory computing: why? and how? *Proceedings of the 6th Conference on Australasian Computing Education (ACE2004), 30*, 255-260.

Raggi, V.L. & Chronis, A.M. (2006). Interventions to address the academic impairment of children and adolescents with ADHD. *Clinical Child and Family Psychology Review, 9*(2), 85-111.

Rapoport, R.N. (1970). Three dilemmas in action research. *Human Relations, 23*(6), 499-513.

Reed, D. (2002). The use of ill-defined problems for developing problem-solving and empirical skills in CS1. *Journal for Computing Sciences in Colleges, 18*(1), 121-133.

Rief, S.F. (1993). *How to reach and teach ADD/ADHD children. Practical techniques, strategies, and interventions for helping children with attention problems and hyperactivity.* West Nyack: Center for Applied Research in Education.

Rief, S.F. (1997). *The ADD/ADHD checklist. An easy reference for parents and teachers.* Paramus: Prentice-Hall.

Robins, A., Rountree, J. & Rountree, N. (2003). Learning and teaching programming: a review and discussion. *Computer Science Education, 13*(2), 137-172.

Ross, E. & Ross, E.C. (2006). The identification of ADHD. *Infants and Young Children, 19*(2), 164-167.

Sá, J. (2002). Diary writing: an interpretative research method of teaching and learning. *Educational Research and Evaluation, 8*(2), 149-168.

Sankaran, S. (1997). Memos to myself: a tool to improve reflection during an action research project. *Action Research Electronic Reader*. Retrieved March 10, 2006 from http://www.scu.edu.au/schools/gcm/ar/arr/arow/rshankar.html

Santrock, J.W. (2005). *A topical approach to life-span development. 6: Cognitive developmental approaches.* Retrieved June 7, 2005 from http://www.uta.edu/psychology/faculty/hallman/classnotes/Santrock%20Chapter%206.ppt

Savitz, J.B. & Jansen, P. (2005). Mainstream and remedial school attention deficit hyperactivity disorder boys: more alike than different. *South African Journal of Psychology, 35*(1), 73-88.

Sayal, K., Goodman, R. & Ford, T. (2006). Barriers to the identification of children with attention deficit/hyperactivity disorder. *Journal of Child Psychology and Psychiatry, 47*(7), 744-750.

Schunk, D.H. (2004). *Learning theories. An educational perspective* (4th ed.). Upper Saddle River, NJ: Pearson.

Shore, K. (1998). *Special kids problem solver: ready to use interventions for helping all students with academic, behavioral and physical problems*. Paramus, New Jersey: Prentice-Hall.

Smelter, R.W., Rasch, B.W., Fleming, J., Nazos, P. & Baranowski, S. (1996). Is attention deficit disorder becoming a desired diagnosis? *Phi Delta Kappan, 77*(6), 429-432.

Sommer, R. (1987). An experimental investigation of the action research approach. *Journal of Applied Behavioral Science, 23*(2), 185-199.

Sunal, D.W. (2007). *The learning cycle: a comparison of models of strategies for conceptual reconstruction: a review of the literature.* Retrieved August 21, 2007 from http://astlc.ua.edu/ScienceInElem&MiddleSchool/565LearningCycle-ComparingModels.htm

Szabo, L. (2006, February 15). Mixed messages on ADHD. *USA Today*, p. 13b.

Thompson, A.M. (2006). Attention deficit hyperactivity disorder: a parent's perspective. *Phi Delta Kappan, 77*(6), 432-436.

Tobin, K. & Tippins, D. (1993). Constructivism as a referent for teaching and learning. In K. Tobin (Ed.), *The practice of constructivism in science education* (pp. 3-21). Hillsdale, NJ: Lawrence Erlbaum.

Tsai, M.-J. & Tsai, C.-C. (2003). Student computer achievement, attitude, and anxiety: the role of learning strategies. *Journal of Educational Computing Research, 28*, 47-61.

Turner, B.A. (1983). The use of grounded theory for the qualitative analysis of organizational behaviour. *Journal of Management Studies, 20*(3), 333-348.

Tytler, R. (2002). Teaching for conceptual change: constructivist/conceptual change teaching approaches. *Australian Science Teachers' Journal, 48*(4), 30-35.

Van Gorp, M.J. & Grissom, S. (2001). An empirical evaluation of using constructive classroom activities to teach introductory programming. *Computer Science Education, 11*(3), 247-260.

Von Glasersfeld, E. (1992). A constructivist's view of learning and teaching. In R. Duit, F. Goldberg & H. Niedderer (Eds.), *Research in physics learning: theoretical issues and empirical studies* (pp. 29-39). Kiel: IPN [Institute for Science Education].

Von Glasersfeld, E. (1995). A constructivist approach to teaching. In L.P. Steffe & J. Gale (Eds.), *Constructivism in education* (pp. 3-15). Hillsdale, NJ: Lawrence Erlbaum.

Wadsworth, B.J. (1996). *Piaget's theory of cognitive and affective development. Foundations of constructivism* (5th ed.). White Plains, NY: Longman.

Weaver, C. (1994). Understanding and educating students with attention deficit hyperactivity disorders: towards a systems-theory and whole language perspective. In C. Weaver, (Ed.), *Success at last! Helping students with AD(H)D achieve their potential* (pp. 1-23). Portsmouth: Heinemann.

Westwood, P. (2003). *Commonsense methods for children with special educational needs. Strategies for the regular classroom* (4th ed.). London: RoutledgeFalmer.

Wilding, J. (2005). Is attention impaired in ADHD? *British Journal of Developmental Psychology, 23*, 487-505.

Williamson, G. & Prosser, S. (2002). Illustrating the ethical dimensions of action research. *Nurse Researcher, 10*(2), 38-49.

Wolraich, M.L. (2006). Attention-deficit/hyperactivity disorder. Can it be recognized and treated in children younger than 5 years? *Infants & Young Children, 19*(2), 86-93.

Working with the ADHD student. (2006). *Techniques, www.acteonline.org*. January, 8-9.

Writing @ CSU: Writing guides. (2005). *Overview: reliability and validity.* Retrieved July 4, 2005 from http://writing.colostate.edu/references/research/relval/pop2a.cfm

Wulf, T. (2005). Constructivist approaches for teaching computer programming. *Proceedings of the 6th Conference on Information Technology Education SIGITE'05*, 245-248.

Appendix A

PO Box [redacted]
[redacted]

19 April 2006

Mr [redacted]
[redacted]

Dear Mr [redacted]

PERMISSION TO UNDERTAKE RESEARCH

As part of my MSc studies at the University of South Africa (in computing education), I would like to conduct a piece of action research into how I can help children with ADHD learn computer programming. I would be grateful if you would give your permission and support for this project.

The work would be done as an afternoon activity and would not interfere with the academic school day in any way. My data collection methods will include field notes and diary recordings. I guarantee that I will observe good ethical conduct throughout. I will secure permission to work with the children from the children and their parents/guardians. I guarantee confidentiality of information and promise that no names of school, colleagues, or children will be made public without your permission and the permission of those who wish to be named.

I promise that I will make my research report available to you for scrutiny before it is submitted, if you wish, and I will make a copy of the report available for your files on its completion.

I would be grateful if you would sign and return the slip below at your earliest convenience.

I enclose two copies of this letter. Please retain one copy for your files.

Yours sincerely

Colin Pilkington

To whom it may concern

I, [redacted], principal of [redacted], give my permission for Colin Pilkington to undertake his research in his classroom and in the school.

_____
[redacted]

Appendix B

PO Box [redacted]

21 April 2006

«Title» «First_Name» «Last_Name»
«Address_Line_1»
«Address_Line_2»
«City»
«ZIP_Code»

Dear «Title» «Last_Name»

PERMISSION TO UNDERTAKE RESEARCH

As part of my MSc studies at the University of South Africa (in computing education), I would like to conduct a piece of action research into how I can help children at a school for children with special educational needs learn computer programming. I would be grateful if you would give your permission and support for «Child» to take part.

The work would be done as an afternoon activity and would not interfere with the academic school day in any way. My data collection methods will include field notes and diary recordings. I guarantee that I will observe good ethical conduct throughout. I guarantee confidentiality of information and promise that I will not reveal the name of the school, colleagues, parents, or children at any time, unless you inform me in writing that you wish me to do so. If you wish, I will keep you informed of progress throughout.

I would be grateful if you would sign and return the slip below at your earliest convenience. Please feel free to contact me should you be at all unclear as to the nature of the research.

I enclose two copies of this letter. Please retain one copy for your files.

Yours sincerely

Colin Pilkington

To Colin Pilkington

I, «Title» «First_Name» «Last_Name», give my permission for «Child» to take part in your research.

_____
«First_Name» «Last_Name»