Concept Mapping and Appropriate Instructional Strategies in Promoting Programming Skills of Holistic Learners

MARIA JAKOVLJEVIC
University of the Witwatersrand, South Africa

The current methods for teaching programming skills fail to concentrate on the effective tools and strategies conducive to the creation of in-depth understanding of programming concepts and procedures. Mind tools such as concept mapping are not appropriately investigated in the present programming classrooms. Thus the needs of learners, particularly the holistic learners, are neglected.

The aim of this paper is to explore concept mapping and instructional strategies applicable in promoting programming skills of holistic learners. The nature of this research required a qualitative research approach where individual interviewing was used to gather data. Ten students at the institution for higher education were identified as holistic learners with a pre-screening device.

The findings of this study reveal the necessity for concept mapping and a peer-based collaborative environment in order to enhance a meaningful programming atmosphere. Holistic learners commented that concept mapping enhanced their learning, as it stimulated simultaneous, deductive or intuitive, concrete, and subjective processes relevant to their style of learning. Concept mapping enabled learners to focus on fine details, experiencing a structured step-by-step approach, representing their knowledge structures graphically and visualising programming concepts and procedures as a network of interrelated ideas. Findings indicated a need for concept mapping to be used in conjunction with practical and cognitive apprenticeship. This will enlarge and strengthen the insight into the programming process and set a climate for enhancing programming skills of holistic learners.

Category and Subject Descriptors**:** K.3 [Computers in Education]: K.3.1Computer use in education – *collaborative learning*, K.3.2 Computers and Information Science Education – *Information Systems and Education;* H [Information Systems]: H.1.2 User/Machine systems – *Human factors, human information processing*
General Terms**:** Design, Human Factors
Additional Key Words and Phrases**:** holistic learners**,** programming skills, concept mapping, instructional strategies, practical and cognitive apprenticeship, peer-based learning, collaborative learning

1. INTRODUCTION

The present instructional methodologies in programming classrooms do not raise awareness of the importance of reflecting on, and exploring a variety of instructional strategies and graphical means in promoting programming skills of learners [Harel and Papert, 1990; Jonassen, 1996]. Holistic learners are divergent thinkers who have difficulties in seeing the whole picture of a subject [Kolb, 1976]. They prefer undirected learning styles, which lead to difficulties in selecting and analysing material, and making decisions [Vermunt, 1996]. Sequential processing of information during programming activities in current programming classrooms results in the overloading of holistic learners' information processing capabilities.

The experience of the researcher reveals that current instructional methodologies in programming classrooms do not support a variety of visual and verbal screen tools. These tools can enable holistic learners to represent programming concepts and label the relationships between concepts, forming a network of ideas which stimulate visual learning. Learners use different flowcharts in mapping the program logic, which have no influence on a deeper understanding of programming aspects.

It is important to consider that learners should be taught how to integrate new knowledge with what they already know, by constructing concept maps through both constructivist and behaviourist instructional approaches [Jakovljevic, 2002]. It is generally accepted that in addition to knowing programming concepts (declarative knowledge) learners must form plans, solve problems and make decisions (procedural knowledge). Concept mapping helps learners to extend their declarative and procedural knowledge by promoting structured knowledge [Diekhoff, 1983; Jonassen, 1996].
It is assumed that the explicit description of programming concepts and their interrelationships through concept mapping will be enhanced within peer-based collaborative learning. Previous researchers [Arzarello, Chiappini, Lemut, Marara, and Pellery, 1993:284] indicate that teaching programming through practical and cognitive apprenticeship can enhance programming skills.

Very little has so far been evident in literature on the creation of a learning environment that will engage holistic learners in the development of programming skills using present instructional strategies and concept mapping as a mind tool.

The purpose of this paper is to explore holistic learners' thoughts and feelings with regard to instructional strategies and concept mapping in programming classrooms. Learners' programming experiences that originated in the programming classroom were explored, analysed and compared with literature findings on concept mapping and current instructional methodologies.

The research questions addressed in this paper are:

What are the strategies that learners perceive as adequate for acquiring programming skills?
How do learners perceive concept mapping as a mind tool in learning programming concepts?

In considering these questions, the researcher argues for more emphasis on in-depth research concerning concept mapping and instructional strategies appropriate to facilitate programming insight and skills of holistic learners. The process aspects of teaching programming skills [Arzarello, et al., 1993] and underlying ideas of mapping knowledge structures of learners [Jonassen, 1996] in the programming context need to receive attention.

The theoretical framework was based on the investigation of computer programming, present instructional strategies, discussing specific nature of the holistic learning style and the essential features of concept mapping.

2. THE THEORETICAL FRAMEWORK FOR THE FACILITATION OF PROGRAMING SKILLS OF HOLISTIC LEARNERS

The following literature review is intended to introduce computer programming, current instructional strategies, attributes of a holistic learning style and concept mapping in order to devise a framework necessary for discussing the findings of this study.

2.1 Computer Programming in the Information Technology (IT) Learning Area

Computer programming is an essential part of the Software Development Life Cycle (SDLC). Mastering programming is a crucial need in the Information Technology (IT) learning area. SDLC is the structured sequence of operations required to conceive, develop and build information systems (IS) [Powell, 199; Harris; 1999; Shelly, Cashman and Rosenblatt, 2001]. Computer programming includes the following stages: understanding the problem, designing and planning the program, coding the program (using commands of a programming language), and comprehending and debugging the program [Pea and Kurland, 1984].

Computer programming is perceived by educators as a time-consuming tool and a complex task requiring months to practice [Harel and Papert, 1996:26]. Jonassen [1996:223] suggests that computer programming is a quasi-mind tool because of its complexity and the need for extensive mental effort.

Taylor [1991] specifies the following five stages in computer programming: problem definition, algorithm design, code writing, debugging and documentation. Algorithm design and transferring algorithms into a computer program are difficult tasks to teach and learn [Taylor, 1991].

2.2 Instructional Strategies In Programming Classrooms

It is generally accepted that it is difficult to convey the method for transforming an abstract statement of a potential computer application into a working problem statement. Programming materials are normally produced in a sequential order. Graham [2002] recommends teaching programming in pairs with two distinct roles: navigator and coder. Learners exchange the roles during programming exercises. Peers provide help through practical and cognitive apprenticeship and tutoring in general [Arzarello, et al. 1993].

Arzarello, et al., [1993] suggest that teaching programming is a cognitive apprenticeship, an interaction between an expert and a novice aimed at enhancing the cognitive and metacognitive skills of learners. They also distinguish between practical apprenticeship and cognitive apprenticeship. Practical apprenticeship is based on three phases: observation, scaffolding and increasingly independent practice. It is aimed at reproducing, possibly with some changes, actions from a model usually given by an expert.

Research supports collaborative work in solving programming problems and designing algorithms, in both educational and corporate settings [Jonassen, 1996:213; Shelly, Cashman and Rosenblatt, 2001: tk3.2; Harel and Papert, 1990:3]. Due to the complexity of skills and outcomes required, most computer projects rely on a vast amount of information, so multiple perspectives are beneficial for computer programming activities [Harrel and Papert, 1990].

The experience of the researcher reveals that holistic learners have problems grasping events and objects during computer programming activities, because the principle of 'association by contiguity' (continuity of events and objects in time and space] proposed by Mayer [1992:12] is not satisfied.

2.3 Conceptualising a Holistic Learning Style

Learning is conceived of as a reorganisation of the learner's knowledge structure, building new structures by constructing new nodes and connecting them with existing nodes [Jonassen, 1988:13]. This view on learning is based on constructivist theory on learning and instruction. Behaviourist theory on learning and instruction point out that the learning occurs as behaviour changes, as a result of guided instruction and reinforcement without taking into consideration the learners' structure of knowledge or mental operations or individual differences in aptitude, expectations, processing abilities, and processing preferences [Winn, 1990:55].

The way individual learners react to the overall learning environment make up individual learning style [James and Gardner, 1995:19]. These authors identify a model of learning styles, which includes three distinct and interconnected dimensions: the perceptual (physiological or sensory) mode, the cognitive (mental or information processing) mode and the affective (emotional or personality characteristics) mode. The cognitive dimension, which involves information-processing habits, is particularly discussed in the literature and distinguished as global (holistic) and analytical learning style [James and Gardner, 1995:20]. According to James and Gardner [1995:21] holistic learners prefer a broad overview of the subject. They state

> 'learners learners using a global approach favour simultaneous, deductive or intuitive, concrete, and subjective processes …the analytical group seeks sequential, inductive, abstract, and objective processes'.

Ehrman [1990:15] argues that the affective (psychological) type affects the preferred learning strategies available to a learner. As the affective dimension must be attended to in every learning transaction [James and Gardner, 1995:21] this is also essential for facilitating programming skills of holistic learners. Whether the affective dimension will be more attended to through concept mapping and whether holistic learners prefer to learn programming with concept mapping is a necessary aspect to consider in promoting programming skills.

Analytical learners perform better in programming classes [Jonassen and Grabowski, 1993] reflecting logical reasoning and direction-following skills [Foreman, 1988]. Mixing analytical and holistic learners in groups enhances critical thinking [Ennis, 1989:5; Jonassen and Grabowski, 1993; cited by Jonassen, 1996:38].

Learners who use holistic approaches are global-dippers [Carnwell, 2000:1]. According to this author these learners tend to be passively engaged with the materials, which leads to surface learning. They need guidance by a tutor, specific instructions, dialogues and closed material. [Sternberg, 1990]. Holistic learners are intuitive and impulsive, forming global impressions by scanning materials [Schmeck, 1988].

Some of characteristics of holistic learning are poor organisation and unclear conception of boundaries [Cooper, 1997; cited by Carnwell, 2000:10]. If holistic learners perceive the structured materials as a 'programme of learning' they may change their preferred learning style to serialist approach [Carnwell, 2000].

Researchers agree on the positive influence of concept mapping as a mind tool to develop meaningful learning and enhance problem solving skills of learners in any learning area [Jonassen, 1996; Kozma, 1987].

2.4 Concept Mapping

Research findings support the use of concept mapping as a mind tool in any learning context [Kozma, 1987; Kommers, 1989; Jonassen, 1996]. Mind tools are computer tools that are intended to engage and facilitate cognitive processing of learners [Kozma, 1987; Kommers, Jonassen and Mayes 1992; Jonassen, 1996]. We need tools 'for depicting and

displaying appropriate knowledge structures, as well as mappings that structure onto the learner's knowledge structure' [Jonassen, 1988:13].

Concept mapping is a metalearning strategy based on the Ausubel-Novak-Gowin theory of meaningful learning [Ausubel, Novak and Hanesian, 1978; cited by Roth and Roychoudhury, 1993:505]. Concept mapping helps learners to reflect on their knowledge, thus emphasising the constructive nature of learning particularly when students collaboratively construct their maps [Roth and Roychoudhury, 1993].

While learning programming, a programmer relates an outstanding concept to its subordinates whether a concept represent a process (e.g. transferring parameters between functions) a procedure (e.g. steps executed in a loop] or a product (e.g. a total number of transactions) [Roth and Roychoudhury, 1993]. Concepts should be connected with linked words, otherwise they are difficult to read.

Instead of a representation that corresponds directly to a network of inter-linked concepts, rather a combination of a hierarchical and network structure is proposed [Afamasaga-Fuata'I, 1998]. This type of concept mapping corresponds to a logical structure of programming knowledge concepts. Learners arrange concepts not only from general to specific as Roth and Roychoudhury [1993] suggest, but also in a logical order following an input processing and output layout.

Concept mapping may be used to identify conceptual understanding of programming concepts. Also concept mapping can be used to represent understanding of an area of knowledge [Thomson, 1997:97]. It can be used as a planning tool and assessment tool for the teacher and as a means of collaborative sharing of knowledge [Thomson, 1997:97].

Learners have the opportunity to acquire structural knowledge through concept mapping [Jonassen 1993; Diekhoff, 1983] which lays a basis for problem solving as an important component in any programming activity [Mayer, Dyck and Viberg, 1986]. Learners interrelate the ideas in a content domain, label and describe the relationships among ideas, which improves their understanding and critical thinking [Jonassen, 1996: 93]. The fact that experts have different structural knowledge compared to novice learners [Jonassen, 1993] suggests an idea that they can map their expert knowledge using concept mapping which may enhance learners' knowledge structure in programming environments.

Presenting computer-programming concepts in the form of a concrete model, as suggested by Mayer, [1992:71], could contributes to learners' understanding of problems during the development of simple computer programs. Engaging learners in concept mapping could satisfy the need for concrete representations of programming concepts.

Based on the literature it seems that the following aspects contribute to meaningful learning in programming classrooms:

— A collaborative learning environment with a mixture of analytical and holistic learning styles
— Multiple dimensions in the programming context: the perceptual, affective and cognitive
— Concrete representations of programming concepts and procedures through concept mapping
— Positive influence of cognitive apprenticeship and activity-based practice.

Although concept mapping has been proved as an effective mind tool in different subjects, supporting the development of structured knowledge as a basis for problem solving, its applicability in programming classrooms need further investigation.

3. RESEARCH METHODOLOGY

This research can be described as descriptive, exploratory, seeing that the learning experiences of students are being investigated relating to a specific context [Yin, 1994; Creswell, 1994; Merriam, 1998].

One distinct mixed cultural group of learners was identified as holistic by using the Index of Learning Style instrument [University of North Carolina, 2000]: ten third-year learners (6 females and 4 males – average age of 20) at an institution of higher education in Johannesburg. This instrument was initially administered to a group of 35 learners.

Participants presented a purposive convenient sample. Learners were taught programming through a variety of instructional strategies (for example, cognitive apprenticeship, peer-based teaching, collaborative learning) as well as concept mapping.

The learners evaluated the interventions in the programming classroom. Ten individual, semi-structured interviews were conducted at the institution of higher education. The researcher conducted the whole process of interviewing. The data analysis in this study is based on a constant comparative method which include the comparison of data within interviews and between interviews [Merriam, 1998].

4. RESULTS

These findings reflect learners' feelings and thoughts regarding concept mapping and instructional strategies (see Figure 1).

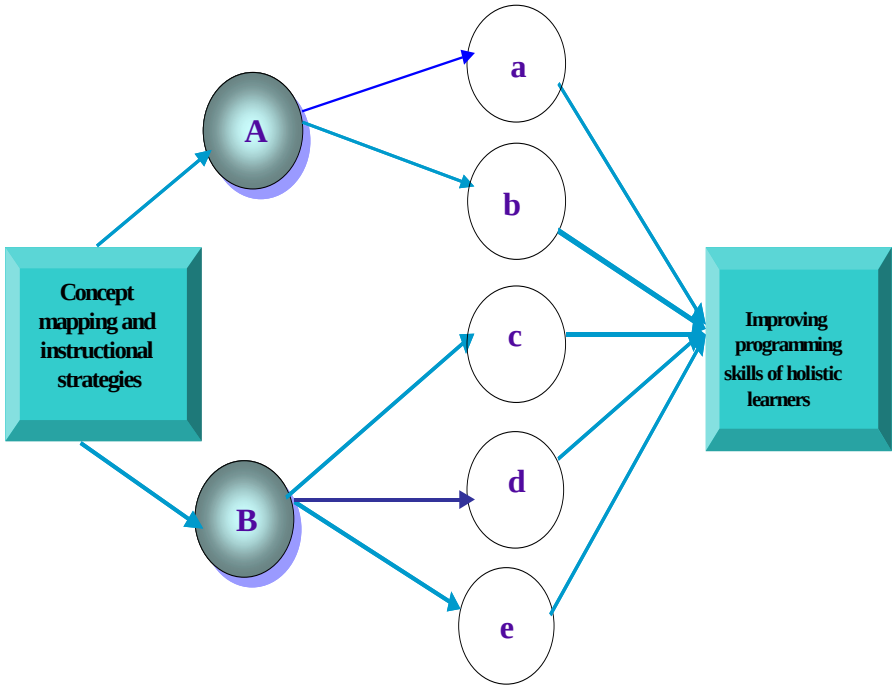

Figure 1.: Categories and sub-categories that relates to learners' experiences in programming classrooms

The letters in the above diagram [Figure 1] represent the following categories and sub-categories:

A. Concept mapping creates opportunities for learners to acquire programming skills

a - *Concept mapping as a planning and organising tool*

b - *Concept mapping as a structured step-by-step process*

B. Collaborative and peer-based learning supported by practical and cognitive apprenticeship provide an essential foundation for developing learners' programming skills

c - *Peer-based learning and activity-based practice*

d - *Collaborative learning*

e - *Practical and cognitive apprenticeship.*

A. Concept mapping creates opportunities for learners to acquire programming skills

a - *Concept mapping as a planning and organising tool*

Learners reported during the interview: '*…it is easier to see the whole process of programming… I can plan a program and links… I can recall ideas …loops… It was clearly presented…I enjoyed like a big program and then when it is broken down into little pieces. We can see the real application of code… an overview of the system'.*
Learners further commented: '*…everyone understands better… It is visual and kind of easy graphics ... That is the only way it sticks in my head… it is such a new way of getting your stuff…'*

b - *Concept mapping as a structured step-by-step process.*

Learners further reported in the interview that concept mapping is a systematic step-by-step process in learning programming: '*It was like a step-by-step process and in the end you could see your final product… If we had more examples with more maps…then we can understand actual programming*.'

Learners could visualise the whole process of programming, recall information, and follow paths in a repetitive, cyclic mode of learning. Concept mapping provided the opportunity to map their knowledge structure and get insight into the basic features of programming.

B. Collaborative and peer-based learning supported by practical and cognitive apprenticeship provide an essential foundation for developing learners' programming skills

c - Peer-based learning and activity-based practice

Learners commented on the usefulness of peer-based learning: *'Amongst peers there is just an easy atmosphere…it is like more easy to talk…you could show others little things…it was easier to explain to each other…everyone understands better.'*

d - Collaborative learning

Different perceptions were recorded concerning learning programming in collaborative groups: '*There is always like someone there to help you out…we discuss it… you can ask questions … If everyone puts effort together and is willing to be there…then it is good…different people grasped different concepts…'*

e - *Practical and cognitive apprenticeship.*

Learners reported the need for continuous guidance and the teacher help: '*Then whoever needed help, the teacher would help them… this is how you do…more practical'*.

Learners further commented: '*We helped each other… John knows programming … they had experience in programming …it was a great relief for our group…'*

Practical and cognitive apprenticeship can reduce learners' frustrations, which positively contributes to learners' motivation and interest with regard to programming tasks. Holistic learners expressed the need for sharing workloads and responsibilities in an emotionally supportive, collaborative learning environment.

5. DISCUSSION

Holistic learners prefer a broad overview of the subject [James and Gardner, 1995:21]. In creating concept maps, learners get an overview of programming concepts and relationships between these concepts. This can produce a good conceptual overview of a system and helps learners to remain on track [Harris, 1999].

Programming processes can be broken into logical sections through concept mapping, and presented in the form of a network [Roth and Roychoudhury, 1993; Afamasaga-Fuata'I, 1998] which resembles human information processing. In this way, the sections of programming code are logically linked and self-explanatory, which assists learners' information processing.

Concept mapping supports deductive learning [James and Gardner, 1995:21] as learners can create an overview of all elements in the visual field, and they have the option to further investigate each element in the field. This can satisfy the 'association of contiguity' [Mayer, 1992] by involving learners in creating links between concepts in the visual field. In addition, holistic learners prefer subjective processes, which can be realised through concept mapping as they map their own knowledge structure. Thus, t he features of concept mapping can fulfil the learning needs of holistic learners for simultaneous, deductive or intuitive, concrete, and subjective processes.

Hill [1998:217], Doornekamp and Streumer [1998:65] support structured learning. DeLuca, [1992:29] remarks that a teacher should establish a sequence of instructions that will lead learners to independent thinking. Holistic learners

seemed to enjoy step-by-step structured guidance, which involved a detailed, repetitive mode of learning through concept mapping. The incorporation of structured guidelines makes it easier to select and sequence learning experiences as one of the procedures relevant for cognitive apprenticeship [Johnson, 1997:174].

While sharing the problem situation in collaborative environments, learners develop problem solving and thinking techniques [Slavin, 1983; Hennessy, Murphy and McCormick, 1994; Wheatley, 1991:19] which are essential for programming tasks. Collaborative learning includes shared knowledge and understanding [Greeno, 1989:139] in an emotionally supportive context [Reid and Stone, 1991:9].

Guiding holistic learners through the programming process with the aid of concept mapping can help them to create a 'concrete model' of the programming concepts proposed by Mayer [1975] which is often not seen in programming classrooms. This process is empowered through cognitive apprenticeship, which supports experiential learning [Kolb, 1984]. With inadequate technological knowledge, learners need modelling, coaching, and scaffolding [Johnson, 1997:175] in order to develop programming skills.

Perhaps holistic learners should be guided through a methodology such as goal-setting, inquiry techniques, assimilation of concepts or ideas, classification, which includes clarification and discussion, activity-based practice and demonstration of accomplishment [Maley, 1978:5].

6. IMPLICATION FOR INSTRUCTION

Instead of providing information through the lecture expository device, as suggested by Smallwood [1995:5], the teacher can improve holistic learners' understanding of programming procedures through concept mapping as it is preferred by learners. Thus perceptual, affective and cognitive dimensions are considered in the programming context.

The need for peer-based collaborative learning in the form of practical and cognitive apprenticeship is confirmed in both interviews and the literature. Research findings support the idea of providing opportunities for learners to discuss the topic with either a tutor or experts [Laurillard, 1994:171; Schon, 1987]. Practical and cognitive apprenticeship could help learners improve programming skills and self-motivation in a peer-based learning environment.

The findings support the virtues of concept mapping as a planning and organisational tool. This tool must be aided by specific instructional strategies for improving holistic learners' programming skills.

In conclusion, the findings from this study can be used for further investigation of concept mapping and diverse instructional strategies for improving programming skills of holistic learners.

7. REFERENCES

AFAMASAGA - FUATA'I, K. 1998. Learning to solve mathematical problems through concept mapping and Vee mapping. National University of Samoa, Samoa.

ARZARELLO, F., CHIAPPINI, G.P., LEMUT, E., MARARA, N., AND PELLERY, M. 1993. Learning Programming as a Cognitive Apprenticeship through Conflicts. In Lemut, E., Du BOULAY, B & DETTORI, G 1993: Cognitive models and intelligent environments for learning models. Germany: Springer-Verlag, 284-297.

AUSUBEL., DP., NOVAK, J.D., AND HANESIAN, H. 1981. *Educational Psychology: A cognitive view*. Second Edition, Rhinehart and Winston: New York.

CARNWELL, R. 2000. Approaches to study and their impact on the need for support and guidance in distance learning. *Open Learning*. 15 (2), 1-16.

COOPER, R. 1997. Learning styles and staff development. *Journal of National Association for Staff Development*. 137, 38-46.

CRESWELL, J.W. 1994. *Research Design: Qualitative and Quantitative Approaches.* Sage Publication, Inc. California.

DeLUCA, W. 1992. Survey of Technology Education problem-solving activities. *The Technology Teacher*, 51(5): 26-30.

DIEKHOFF, G.M. 1983. Relationship judgements in the evaluation of structural understanding. *Journal of Educational Psychology*. 75, 227-233.

DOORNEKAMP, B.G., & STREUMER, J.N. 1996. Problem-solving in teaching/learning packages for technology. *International Journal of Technology and Design Education*, 6: 61-82.

EHRMAN, M. 1990. Psychological factors and distance education. *American Journal of Distance Education.* 4, (1), 10-24.

ENNIS, R.H. 1989. Critical thinking and subject specificity: Clarification and needed research. *Educational Researcher*. 18(3), 4-10.

FOREMAN, K.H.D. 1988. Cognitive style, Cognitive ability, and the Acquisition of Initial Computer Programming Competence. In M. Simonson eds, *Proceedings of selected research papers presented at the annual meeting of the Association for Educational Communication and Technology.*

GRAHAM, S. 2002. Introduction to Java Using Ready. *South African Symposium on computer programming.* 19 – 20 September, Johannesburg.

GREENO, J.G. 1989. A perspective on thinking. *American Psychologist*. 44(2), 134-141.

HAREL, I., AND PAPERT, S. 1990. Software design as a Software design as a learning environment. *Interactive Learning Environments*. 1, 1-32.

HARRIS, D. 1999. *Systems Analysis and Design for the Small Enterprise.* The Dryden Press, Harcourt Brace College Publishers. Fort Worth.

HENNESSY, S., MURPHY, P., AND McCORMICK, R. 1994. The General Problem Solving Process in Technology Education: Myth or Reality? *International Journal of Technology and Design Education*, **4**(1), 5-34.

HILL, A.M. 1998. Problem-solving in real life contexts: An alternative for design in Technology Education. *International Journal of Technology and Design Education*. 8, 203-220.

JAKOVLJEVIC, M. 2002. An Instructional Model for Teaching Complex Thinking through Web Page Design, [DEd thesis]. Rand Afrikaans University, Johannsburg. South Africa.
JAMES, W.B., AND GARDNER, D.L. 1995. Learning styles: Implications for distance learning. *New directions for adult and continuing education*. 67,19-31.
JOHNSON, S.D. 1997. Learning technological concepts and developing intellectual skills. *International Journal of Technology and Design Education*. 7,161-180.
JONASSEN, D.H. 1988. Designing structured hypertext and structuring access to hypertext. *Educational Technology*. Nov, 13-16.
JONASSEN, D.H. 1996. *Computers in the Classroom: Mind Tools for Critical Thinking*. Prentice-Hall. New Jersey.
JONASSEN, D.H., AND GRABOWSKI, B.L., 1993. *Handbook of individual differences, learning and instruction*. Erlbaum Associates. Hillsdale, N.J.
KOLB, D.A., 1976. *Learning style inventory self scoring test and implementation booklet*. McBer and Company. Boston.
KOLB, D.A. 1984. E*xperiential learning: experience as the source of learning and development*. Prentice-Hall. Englewood Cliffs.
KOMMERS, P.A.M. 1989. *TextVision*. Enschede, Netherlands: University of Twente, Faculty of Education.
KOMMERS, P., JONASSEN, D.H., AND MAYES, R. 1992. *Cognitive tools for learning*. Heildeberg, Germany: Springer-Verlag.
KOZMA, R.B. 1987. The implications of cognitive psychology for computer-based learning. *Educational Technology*. 24 (11), 20-25.
LAURILLARD, D. 1994. Rethinking University Teaching: A framework for the effective use of educational technology. Routledge. London.
MALEY, D.A. 1978. *The Industrial Arts Teacher's Handbook: Techniques, Principles and Methods.* Allyn and Bacon, Inc. Boston, MA.
MAYER, R.E. 1975. Different problem-solving competencies established in learning computer programming with and without meaningful models. *Journal of Educational Psychology*. 67, 25-734.
MAYER, R.E. 1992. *Thinking, Problem Solving, and Cognition.* W.E. Freeman and Company. New York.
MAYER, R.E., DYCK, J.L., AND VIBERG, W. 1986. Learning to program and learning to think: What's is the connection? *Communications of ACM*. 29[7], 605-610.
MERRIAM, S.B. 1998. *Case Study Research in Education: A Qualitative Approach.* Jossey-Bass. San Francisco.
PEA, R.D., AND KURLAND, D.M. 1984. On the cognitive effects of learning computer programming. *New Ideas in Psychology*. 2 (2), 137-168.
POWELL, T.A. 1999. *The Complete Reference: HTML* (2nd edition). Osborne/McGraw-Hill. Berkeley.
REID, D.K., AND STONE, C.A. 1991. Why is cognitive instruction effective? Underlying learning mechanisms. *Remedial and Special Education [RASE]*. 12 (3), 8-19. Special issue: cognitive instruction and problem learners.
ROTH, W.M., AND ROYCHOUDHURY, A. 1993. *The concept map as a tool for the collaborative construction of knowledge: A microanalysis of high school physics students*. John Wiley & sons, Inc. Canada.
SCHMECK, R.R., (Ed) 1988. *Styles and Strategies of Learning.* Plenum. New York.
SCHON, D.A. 1987. *Educating Reflective Practitioner. Toward a New Design for Teaching and Learning in the Professions*. Jossey-Bass. San Francisco.
SHELLY, G.B., CASHMAN, T.J., AND ROSENBLATT, H.J., 2001. *Systems Analysis and Design* [4th edition]. Course Technology a division of Thomson learning. Boston.
SLAVIN, R. 1983. *Cooperative Learning*. Longman. New York.
SMALLWOOD, J. 1995. Technology Discussion in the Classroom. In Edmision, G.A., 1995: Delivery systems: Instructional Strategies for Technology Education. International Technology Education Association [ITEA]: Reston VA, pp.5-6.
STERNBERG, R.J. 1990. Thinking styles: keys to understanding human performance. *Phi Delta Kappan*. 71,366-371.
TAYLOR, C. 1991. The Dialogical Self In Hiley, D.R, J.F., Bohman, J.F., Shusterman, R., eds. The Interpretive Turn: Philosophy, Science, Culture. Ithaca: Cornell University Press, pp. 304-314.
THOMSON, C.J. 1997. Concept mapping as a means of evaluating primary school technology programmes. *International Journal of Technology and Design Education*. 7, 97-110.
VERMUNT, J.D. 1996. Metacognitive, cognitive and affective aspects of learning styles and strategies: a phenomenographic analysis. *Higher Education*. 31, 25-50.
WHEATLEY, G.H. 1991. Constructivist perspectives on science and mathematics learning. *Science Education*. 75[1], 9-21.
WINN, W. 1990. Some implications of cognitive theory for instructional design. *Instructional Science*. 19, 53-69.
YIN, K.R. 1994. *Case Study Research – Design and Methods.* Sage Publications. Thousand Oaks: California.