MENTAL MODELS OF ROBOTIC PROGRAMMING

Alfred Mutanga
Management Information Systems, University of Venda, South Africa
Email: alfred.mutanga@univen.ac.za

Abstract
This study details the Mental Models of Robotic Programming which Students used in a Supplementary Programmable Embedded Systems hands-on course. The course was taught using the cognitive apprenticeship instructional methodology. The course was supplemental to an existing Artificial Intelligence Course at Instituto Superior de Humanidades e Tecnologias, of the Universidade Politécnica de Moçambique, in Mozambique. Twelve students participated in the research. Several tools where applied to elicit the mental models of the students. The students' mental models were compared with predetermined robotic programming mental models. The number of correct robotics programming mental models increased during the hands-on activities. However, the study identified a number of correct and incorrect robotic programming mental models used by the students. Some students were found to not constantly using the correct robotic mental models. The reason for this has been attributed to the mental models theory which states that mental models are unstable. The study also revealed that the use of cognitive apprenticeship as an instructional approach helped the students to develop correct robotic programming mental models.

Keywords: Mental Models, Cognitive Apprenticeship, Constructivism

1.1 INTRODUCTION

Students enter into computer science disciplines, especially programming related disciplines, with preconceived ideas of the concepts that they will be engaged in especially when solving hands-on activities (Stelios and Maya, 2005). Previous researchers have found that students are not only motivated by the practical use of computer science concepts, but also, by having tangible artefacts which they have produced on their own. In this context computer science teachers are therefore obliged to design activities that provoke the students' motivation to learn computer science concepts and implement them in practice. Hands-on activities help students to engage more in the learning process (Stelios and Maya, 2005). It had been claimed that through hands-on activities students can easily establish contextual interrelationships between computer science concepts. During instruction teachers have to exhume these students' preconceptions, prepare them to grasp new concepts and guide them into the attainment of academically accepted computer science knowledge and skills (Ma, Ferguson, Roper and Wood, 2007).

The identification of cognitive processes in computer science led to the development of hands-on activities that uncover the students' epistemological issues during the resolution of the activities (Jih and Reeves, 1999). The studies carried out by (Bornat, Dehnadi and Simon, 2008; Jin and Reeves, 2004; Kahney, 1989; Ma, Ferguson, Roper and Wood, 2007; Sanders, Galpin and Gotschi, 2006) of the internal mental representations of the students' cognitive processes were to determine what 'mental models' were being used by the students when they were confronted by a particular computer science concepts. Referring to how students interact with computer programs, Jih and Reeves (1999) said that mental models are internal "*structures reflecting the users' understanding of what the program contains, how it works, how the components are related, what the internal process are, and why the program works in the way it does*" (Jih and Reeves, 1999 , p.45).

Hands-on activities and the students' mental models were important concepts in this research study. We designed, developed and implemented a Supplementary Programmable Embedded systems hands-on course for students in the Information Technology Management degree programme at the Universidade Politécnica de Moçambique, in Mozambique. We principally developed a hands-on based course for two main reasons. The first reason being that the instructional framework emphasized to direct the teaching and learning of all the courses at Universidade Politécnica de Moçambique is based on the polytechnic philosophy (Miquidade& Chinama, 2003). The second being that the Artificial Intelligence course in the Information Technology Management degree programme has been taught theoretically over the years and there were no hands-on laboratory activities for the students. In our view the fact that there were no hands-on activities for the course contradicted the polytechnic characteristic envisaged in the pedagogical philosophy of the university. We presumed that during the learning process the students would build robotic programming mental models based on the programming resources offered by the LEGO NXT-G implementation environment. The students would be creating mental representations of what the robots will do, what actions cause what robotic behaviours and why those actions caused those behaviours (Potosnak, 1989). In actual fact, in this research, we treated the concept of mental models as the various ways in which the students understood the different robotic programming contexts engulfed in the designed hands-on activities. The implementation part of the course used the cognitive apprenticeship instructional methodology to deliver the course.

1.2 THEORETICAL FRAMEWORK

The delivery of computer science concepts to students is mainly based upon two philosophies objectivism and constructivism (van Gorp and Grisson, 2001). According to (van Gorp and Grisson, 2001) objectivists believe that students learn through listening to the teacher explaining, and engaging them in reinforced practice, and respond to external motivation. The assessment of the students' learning in the objectivist approach occurs by measuring observed and quantifiable behavioural outcomes on predefined tasks (van Gorp and Grisson, 2001). Constructivism is a theory of learning which claims that students construct their knowledge rather than merely receive and store knowledge transmitted by the teacher (Ben-Ari, 2004).In this case constructivism is a

contrast to instructional methods were students are mere receptors of the subject matter. Constructivism is an alternative pedagogical approach that focuses more on the learning and experiences of the student (van Gorp and Grisson, 2001). In constructivism the assessment of the students' learning evaluates the functioning of the students as well as their ability to defend and explain decisions through developed metacognitive skills (van Gorp and Grisson, 2001). In constructivism the students learn by actively constructing their own meanings and understanding of the concepts at hand. The students usually achieve learning by connecting to their previous experiences and considering alternative perspectives held by others (Schulte, Magenheim, Niere and Schafer, 2003).

This research focused on cognitive apprenticeship as the pedagogical approach in the teaching of the supplementary, programmable embedded systems, hands-on course for an existing Artificial Intelligence course at Universidade Politécnica. Cognitive apprenticeship teaching methods are bound by the constructivist philosophy. Constructivism is a theory of learning which claims that students construct their knowledge rather than merely receive and store knowledge transmitted by the teacher (Ben-Ari, 2004).In this case constructivism is a contrast to instructional methods were students are mere receptors of the subject matter. Many students find the study of computer science extremely difficult (Ben-Ari, 2004) compared to other disciplines in the arts area. Students construct knowledge by combining the experiential world (Ben-Ari, 2004) with existing cognitive structures. The application of constructivism to computer science education must take into account that a beginning computer science student has no effective model of a computer and the computer forms an accessible ontological reality, (Ben-Ari, 2004).

The term 'cognitive apprenticeship' was first publicised by (Collins, Brown and Newman, 1989). Some of the definitions of the cognitive apprenticeship that come out from the literature are:

1. *Cognitive apprenticeship is an instructional design model which is based on the current understandings of how individuals learn* (Ghefaili, 2003).

2. *Cognitive apprenticeship is a method of teaching aimed primarily at teaching the processes that experts use to handle complex tasks* (Collins, Brown and Newman, 1989).

The theory of cognitive apprenticeship holds that masters of a skill often fail to take into account the implicit processes involved in carrying out complex skills when teaching novice (Collins, Brown and Holum, 1991). The definition of cognitive apprenticeship is therefore bound by the traditional apprenticeship concept and situated cognition. Situated cognition is a theory of instruction that suggests that learning is naturally tied to authentic activity, context and culture (Brown, Collins, and Diguid, 1989). Apprenticeship teaching and learning (Hay and Barab, 2001) has its roots in the pre-industrial age, where an apprentice would work alongside a master of trade. Apprenticeship can be broadened as a legitimate peripheral participation, a generative social practice where a beginner, novice, or newcomer is gradually enculturated, with the goal of becoming an expert (Lave and Wenger, 1991). Situated learning which has its foundations on Vygotsky's theories of learning and child development, claims that children learn from their cultural setting, by internalizing the knowledge and practices of their social environment (Vara and

Tan, 2008). Cognitive apprenticeship allows students to be enculturated in authentic practices and social interaction; as they develop the cognitive skills of the expert practitioner. The cognitive apprenticeship model is comprised of six teaching methods: modelling, coaching, scaffolding, articulation, reflection and exploration.

In **Modelling,** the teacher (cognitive master) models processes to show "how the processes unfolds" or how the mentor/peers function in certain situations (Ghefaili, 2003; Collins, Brown and Newman, 1989; Darabi, 2005). In other words, the teacher performs a task so that students can observe his actions and build a conceptual model of the processes required to accomplish that task. From a cognitive point of view the students have internal (mental) representations of how to solve the problems, but modelling helps in the externalization of the students' internal cognitive processes. In **Coaching**, "the cognitive master (teacher) provides assistance to students as needed by providing individual attention on difficulties the students are having, providing help at "critical times" or when the students most need it, providing requested assistance as needed and withdrawing unneeded help, and asking relevant questions to stimulate thought and provide a different point-of-view of situations" (Ghefaili 2003). During coaching the conceptual and factual knowledge are put into practice and usually they are situated in their contexts of use.

In **Scaffolding**, the teacher helps students to manage a more complex task performance. Here the term complex is relative, because one task that might be complex to one student and might not be that entire complex to the other. The teacher is therefore supposed to identify those parts of the task that students have not yet mastered and provide them with necessary scaffolds to accomplish the tasks. The gradual removal of the teacher's support to the students as they learn to manage the task at hand on their own is called *fading*. In **Articulation**, students come out of their shells to explain, think and give reasons about what they are doing, why they have taken certain decisions and why they have selected certain problem-solving strategies. The articulation method makes the students' tacit knowledge explicit and this helps in incorporating this knowledge in the problem solving strategies.

In **Reflection**, students reflect on the work they have already performed and analyse, criticize it or deconstruct it. Through this process, they can increase their "awareness of their own knowledge" (also called metacognition) and be able to compare what they know with what others know (Darabi 2005; Ghefaili 2003; Cope, 2005; Wang and Bonk 2001). The role of the teacher is to provoke or incite the students to make a comparison of their problem solving processes with his model answer which will have the relevant cognitive expertise. In **Exploration**, students will be trying out different hypotheses, methods and strategies by studying and analysing their projects and their working environment. Students will be thinking of alternative solutions to the tasks at hand and learn how to set achievable goals, form and test hypotheses, and make independent discoveries.

Cognitive apprenticeship has been used in computer science in the teaching-learning of different computer science concepts. The conclusions which Shabo, Guzdial, and Stasko (1997) made are that with different kinds of scaffolding, by communicating the processes to students, coaching and eliciting articulation are useful cognitive apprenticeship strategies to give highly structured support

to the students' learning. Enkenberg (1994) studied the situated programming in a Lego logo environment based on situated programming and cognitive apprenticeship providing the students with the skills to organize procedural paths that have similarities to the behaviour of experts (Enkeberg, 1997). The research of Schulte, Magenhein, Niere and Schafer (2003) in upper secondary schools in Paderborn, Finland showed the positive contribution made by cognitive apprenticeship, in developing correct mental models of object-oriented technology.

Cognitive apprenticeship proved to be a valuable instructional approach as it was evident that the synergy between software engineering and cognitive apprenticeship helped students to understand relationships between software productivity and software quality (Huang, Cho and Ling, 2006). Jarvela (1995) claimed that cognitive apprenticeship learning and research interests have focussed mainly on learning outcomes or teaching strategies. In this research cognitive apprenticeship instruction is used to study the students' mental models in different robotic programming contexts. Mental models are credited to the work of Kenneth Craik in 1943 when he 'proposed the mental models theory as an explanation for human thought process' (Edwards-Leis.2007). Craik defined mental models as "dynamic representations of the reality of a system held by users" and as "small scale models of reality" (Edwards-Leis, 2007). The precise definition of the term 'mental model' is difficult to give because there is no an agreed definition from different scholars. In support of this conjecture Edward-Leis 2007 wrote:

1. *"Definitions of mental models appear to be as idiosyncratic as mental models themselves" p.26*

Zdeslav (2002) also wrote:

2. *"Wider studies of mental model definitions show that no consensus exists about the definition of the term mental model and "some definitions of the concept are even contradictory" p.19*

Lei, Yang and Zhang (2006) wrote:

3. *"Currently there is no universal definition of user's mental models that can be found in existing literatures"*

The definition of mental models for this study is "an internal mental representation, which acts out as a structural analogue of situations or processes" (Greca and Moreira, 2002). The role of mental models in this study is to account for the students' reasoning during the learning process, when they try to reason, explore, articulate and reflect on the learning activities presented in the hands-on course. The study of mental models in computer science does not span a long time. Studies by Sanders, Galpin and Gotschi, (2006) and Kahney (1983), found out that students were having non-viable mental models of the basic programming concepts which they suspected that they were causing misconceptions and problem solving difficulties. Dehnadi, Bornat and Simon (2008, 2009) used a test questionnaire that was covering the programming concepts of assignment (*reference* and *value* assignment) and it captured qualitative data on the students' mental models of these concepts.

In a similar study Ma, Ferguson, Roper and Wood (2007) investigated the viability of students' mental models of reference and value assignment in Java programming, and to study the relationship between the novice programmers' mental models and their performance in examinations and the programming tasks given during the period of the course. Their results indicated that students held a variety of mental models of value and reference assignment. Ma, Ferguson, Roper and Wood, (2007) further argued that effective learning in the constructivist approach incites the construction of viable mental models. Brooks and Scott (2007) determined the mental models variability in the perception and the reasons for this variability of the software engineering methodological tools. They discovered three belief biases for the tools and concluded that the inspection-style approach supports replication of work and they recommended the work for laboratory work for mental models research.

1.3 RESEARCH QUESTIONS, RESEARCH DESIGN AND METHODOLOGY

The study identified the mental models which students use in solving the robotic programming tasks given to them. While it is imperative to have hands-on activities in computer science disciplines, the instructional approaches should support the students in acquiring conceptual, factual and procedural knowledge. Student learning should be naturally tied to authentic activity, context, and culture. We decided to implement robotic programming algorithms, using Lego Mindstorms NXT. We thought that the understanding of the students' mental models in different robotic programming contexts will help us to have a deeper understanding of how the students perform problem solving tasks.

We designed a learning environment, where students were presented with real-world robotic programming hands-on activities which were embedded with authentic tasks that emphasized social interaction and situated learning. In this process we became interested in knowing the mental models of the robotic programming contexts that the students' use in the dialog, reasoning and resolution of the hands-on activities. In the pursuit of providing hands-on activities and trying to deliver the hands-on activities in a constructivist way, the following research questions guided the research study.

RQ1. What are the mental models of robotic programming students use in solving the hands-on activities in the supplementary programmable embedded systems hands-on course?

RQ2. How does the number of correct students' mental models of the robotic programming change during the process of solving the hands-on activities?

RQ3. To what extend do the number of correct mental models of robotic programming go up after using cognitive apprenticeship as a method of instruction?

The philosophical or paradigmatic framework that governed this research study is the *qualitative* research paradigm. The research instruments used in the study were the mental models test questionnaire (MMTQ); think-aloud protocol, teach-back protocol, cognitive interview with the students (audio recorded), Students' programs analysis, and video recordings of the teaching

sessions. The mental model test questionnaire was applied before instruction, during instruction in the fourth session and after instruction in the sixth session (last session). A set of ten questions based on multiple-choice questions constituted the MMTQ. Multiple choice questions have four components which are the *stem* (questions or incomplete statements), *options* (suggested answers or completions), *distracters* (incorrect responses) and the *key* (correct responses) (Woodford & Bancroft, 2005).

The think-aloud protocol was used to make the students verbalize their actions when resolving the hands-on activities. Its use was to make it possible to identify what NXT blocks and sensors students used in the resolution of the hands-on activities and how. The students would speak aloud and verbalize every action they have taken in solving the hands-on activities. The design of the think-aloud protocol questions was closely linked to the heuristic strategies of the cognitive apprenticeship instruction. The teach-back protocol was used as a means of discovering the information about the knowledge the students have about the expected NXT blocks and sensors to be used in the hands-on activities. The design of the teach-back protocol was coupled with the articulation method of the cognitive apprenticeship instruction. Three teach-back sessions were developed with each of the three student groups having teaching a session. In the teach-back session we expected the students to externalize the mental models of the robotic programming they have used in the resolution of the hands-on activities. The '*what'* question to help to elicit the students' conceptual knowledge of the NXT blocks and sensors. The '*how'* question elicited the procedural knowledge that the students used solve the activity.

The cognitive interview was used as a technique to explore the mental processes used by the students in answering the questions about the NXT blocks constituted in our research. We used the cognitive interview as a means of uncovering the covert and overt problems that we might have done in the designing of the MMTQ. Methodologically we employed the verbal probing technique, by interviewing members of each group at the same time. We also designed the cognitive interview based on the articulation and evaluation methods of the cognitive apprenticeship instruction methodology. In actual fact, we triangulated the research instruments to elicit the students' mental models of the NXT programming blocks as a means of establishing the validity our predictions.

1.4 Research Participants and Research Protocol

The research involved twelve fourth year Licentiate degree in Information Technology Management students. There were nine male students and three female students who took part in the research. Since we had three NXT robot kits, we organised students into three groups. One female student was in each of the group. The organisation of the group was based on the performance of the students in previous programming related courses. We made sure that the groups were balanced in terms of the students' performances in those previous programming courses. In terms of infrastructure the research needed LEGO Mindstorms NXT kits and computers capable of running the NXT-G software and satisfying the recommended hardware requirements for Mindstorms NXT. The students had access to a wide range of local area network services. For our research the students had access to the file server, where they could get the robotic building

instructions and also store their NXT program files. We created folders where the students were getting the building instructions for the robots. Each group had its own folder (*Group1, Group2 and Group3*) where they were putting the solutions of each of the NXT activities the group had developed. We configured the security of the group folders in such a way that it only allowed file access to members of the same group.

Each teaching activity took duration of three contact hours and a week for students to develop the robots and present. All the laboratory sessions were videotaped. We developed and delivered the following activities:

- *Day 1 Activity 1: Introduction to LEGO Mindstorms NXT (3 hours)*
- *Day 2 Activity2: Door Alarm (3 hours)*
- *Day 3 Activity 3: The Cricket Batter (3 hours)*
- *Day 4 Activity 4: The Line Follower (3 hours).*
- *Day 5 Activity 5: The Penalty Taker (3 hours)*
- *Day 6 Activity 6: Volume and Area (3 hours)*

1.5 Data Analysis and Results

The credibility of the students' responses depended on the relationships between the data obtained through the think-aloud protocols, cognitive interviews, teach-back sessions, audio and video recordings and the coded categories describing the students' mental models of the different robotic programming contexts. The nature of data analysis revealed the students' conceptions, students' viewpoints of the robotic programming contexts and these could be used to establish the students' mental models categories. We did not want to place a framework upon the students' ideas, but through the cognitive apprenticeship instructional process, we wanted to identify what were the students' mental models and how these change over time. As an example through the data collection instruments we got the following context of the mental model:

Context 1: Effectors Mental Model - using the Move and Motor Blocks

Heuristic Question1: What will be happening to your robot when you change different parameters in the Motor and Move Block configuration panes?

Group1: *These blocks are used for robot motion or navigation. There is no much difference between the move and motor blocks. Actually they perform the same things. We can have the robot move for some predefined revolutions, degrees, coast or stop.*

Do you all agree that there is no difference between the Move and Motor Blocks?

Group 1 *Ummm, Yes*

Group2: *The motor block is used to turn the robot motors on and off. Its action panel can be used to program the robot to move at a constant speed, ramp up or ramp down. The move block allows for more control on how the robot moves, and it can be used to program more of how the motors move.*

Group3*: The Motor block specify how the motors of the robot will behave, and the Move block does not have the Action panel that the motor block has. As a matter of fact the Move Block has built-in algorithms that control the Motor's rotation sensors. The move block can be configured to make the robot move and behave in the desired way.*

1.6 CONCLUSIONS

Our findings reflect that most of the students used the correct robotic programming mental models based on the robotic programming contexts embedded in the hands-on activities. When we designed the six hands-on activities, we predetermined six major robotic programming mental models, which are the actuators/effectors mental models, sensors mental models, program flow mental models, memory management mental models, data flow mental models and the computational mental models. These six major mental models were decomposed further and resulted into ten minor robotic programming mental models that came up as a result of the implementation of the hands-on activities in the Lego Mindstorms NXT environment. To answer the research question:

RQ1. *What are the mental models of robotic programming contexts used by the students in solving the hands-on activities in the supplementary programmable embedded systems hands-on course correct or incorrect?*

The following table indicate which mental models were identified and how many students used the correct mental models.

Major Mental Models	Minor Mental models	Number of Students with Correct Mental Models
Actuator	Move and Motor	12
Program Flow	Loop	12
Program Flow	Switch	9
Memory Management	Variable	11
Computational	Math	12
Data Flow	Data hub and Data wires	2
Program Flow	Wait	7
Sensors	Light Sensor	3
Sensors	Ultrasonic Sensor	12
Sensors	Touch Sensor	12

Table 1: Mental Models elicited during the research

Research question RQ2 thrives to explore how the students' mental models change during instruction of the different robotic programming contexts of embedded in the hands-on activities. Research question RQ2 is stated as:

RQ1. *How does the number of correct students' mental models of the robotic programming change during the process of solving the hands-on activities?*

In answering the research question RQ2 we have looked at the students' mental models at different stages during the delivery of course. Quantitatively the students attained the intended

mental models for the different programming contexts in the supplementary embedded systems hands-on course by the end of the course. Only two students had correct move and motor mental models before instruction, but all the students have the correct move and motor block mental models during and after instruction. The findings from the MMTQ showed that there were students who were constantly using the correct robotic programming mental models and some students were not constantly using the correct robotic programming mental models. Analysing how the students behaved in each of the ten robotic programming mental models we could say that the two students who had correct move and motor mental models were constantly using them correctly during and after cognitive apprenticeship instruction.

Since the instructional approach used in the research intervention was the cognitive apprenticeship approach, it is important to detail whether it had any influence on the mental models. The answers to the research question RQ3 will provide information regarding this respect. Research question RQ3 states:

RQ1. *To what extend do the number of correct mental models of robotic programming go up after using cognitive apprenticeship as a method of instruction?*

The number of correct students' mental models of robotic programming went up during the hands-on activities. To substantiate the yes, we will start by saying the number of students with correct mental models were increasing as reflected by the results from the think-aloud, teach-back protocols and the MMTQ applied during and after instruction. The cognitive apprenticeship and its indicators which we embedded in the data collection instruments also contributed to these changes. For example, the heuristic questions used in the think-aloud protocol were part of the cognitive apprenticeship heuristic strategies indicators. The heuristic questions provoked students to expound their knowledge of the robotic programming contexts verbally and loudly to the whole class. Those explanations and the responses the students given to the heuristic questions fulfilled the cognitive apprenticeship indicators for the articulation method as well.

1.7 DISCUSSION

Our experiences through our research are that mental models as internal mental representations very difficult to elicit and measure. For this reason we had to base on eliciting these through the robotic programming contexts and on the NXT programming blocks. Even having NXT-G programming blocks the mental models that were based were not available for direct measurement. Being confronted with such a dilemma of measuring mental models we had to think of multiple robotic programming mental models measuring tools. The measuring tools were based on verbal protocols which were meant to exhume what the students were actually thinking when solving the hands-on activities and performance tests. The verbal based data collection instruments which assessed the robotic programming mental models which the students used during the course were the think-aloud protocol and the teach-back protocols. The teach-back protocol was representative of the students' robotic programming mental models during the instruction as it was applied when the students were solving the problems.

The findings can be explained by the characteristics of mental models themselves, and following the claims by Norman (1983), Redish (1994) and Zdeslav (2002) mental models contain contradictory elements, they do not have firm boundaries and students may feel uncertain with their knowledge. The arguments raised from the mental models theory might be the reasons why the total number of correct switch mental models and wait mental models were varying during the research process.

However we have some comments about the data collection instruments we have used in our research. First the teach-back protocol, think-aloud protocol, cognitive interview and students program analysis data collection techniques were applied to individual student groups. Each group had four students. The answers we got were contributed by the group so this might not have been the true reflection of the actual robotic programming mental models used by each student. It would have been better to have applied these techniques to individual students and have a true picture of what robotic mental models each student used. This implies having more NXT kits and more teaching time. However in our case we had only three NXT robotic kits and it was not possible to apply these techniques to the individual students.

The previous research in mental models in the computer science disciplines have mostly used observational and regression techniques. Examples of other works which have statistical works include George (2000) who used percentages and interviews; Brooks and Scott (2007) used percentages; Ma, Ferguson, Roper and Wood (2007) used percentages and the Wilcoxon Mann-Whitney Test to compare the groups of students with consistent mental models and inconsistent mental models. Having said this, we understand that, there are methodological considerations to be taken into account to perform regression analysis of the test questionnaires. These include the sample size; how students are involved in the learning process; how many hours the students are taught and the composition of the student population (factors include gender, prior programming experiences etc.). In our case we had twelve students and the p-values, and frequencies do not pose a lot of restrictions with respect to population size as compared to other techniques. We would have wanted to apply the MMTQ to a large sample of student group and apply multiple regression procedures.

1.8 REFERENCES

Ben-Ari, M. (2001). Constructivism in Computer Science Education. *Journal of Computers in Mathematics and Science Teaching, Volume 20,* (Issue 1), pp. 45-73.

Ben-Ari, M. (2004). Situated Learning in Computer science Education. *Computer Science Education, Vol. 14*(2), 85-100.

Bornat, R., Dehnadi, S., & Hamilton, S. (2009). *Mental models, consistency and programming aptitude*. In *Proceedings of the Tenth Conference on Australasian Computing Education - Volume 78* (Wollongong, NSW, Australia, January 01 - 01, 2008). S. Hamilton and M. Hamilton, Eds. Conferences in Research and Practice in Information Technology Series, vol. 315. Australian Computer Society, Darlinghurst, Australia, 53-61.

Brooks, A., & Scott, L. . (2007). *A Methodology from Software Engineering Inspection which Supports Replicable Mental Models Research.* . Paper presented at the Proceedings of the Six IEEE International Conference on Cognitive Informatics ICCI 2007 August 6-8, 2007, Lake Tahoe, CA, USA. .

Brown, J. S., Collins, A., & Duguid, P (1989). Situated Cognition and the Culture of Learning. *Educational Researcher, 18*(1), 32–42.

Collins, A., Brown, J.S., & Newman, S.E. (1989). Cognitive Apprenticeship: Teaching the Craft of Reading, Writing, and Mathematics. . *In L.B. Resnick (Ed.), Cognition and Instruction: Issues and agendas*, pp. 1-37.

Collins, A., Brown, J.S., & Holum, A. (1991). Cognitive Apprenticeship: Making Thinking Visible. *American Educator, Quarter 4*(Winter 1991), pp. 1-18.

Cope, N. (2005). Apprenticeship reinvented: Cognition, discourse and implications for academic literacy. *Prospect, 20*(3), 42-62.

Darabi, A. (2005). Application of cognitive apprenticeship model to a graduate course in performance systems analysis: A case study. *Educational Technology Research and Development (ETR&D), 53*(1).

Douglas, I., Sanders,V., & Galpin., C. (2007). *Students' mental models of recursion at wits*. In *Proceedings of the 12th Annual SIGCSE Conference on innovation and Technology in Computer Science Education* (Dundee, Scotland, June 25 - 27, 2007). ITiCSE '07. ACM, New York, NY, 317-317. DOI= http://doi.acm.org/10.1145/1268784.1268883

Edwards-Leis, C. (2007). Matching Mental Models: The starting point for authentic assessment in robotics. *Design and Technology Education: An International Journal, 12*(2), 25-36.

Enkenberg, J. (1994). Situated programming in a legologo environment. *Computer Science Education, Volume 22* (Issues 1-2), Pages 119-128.

George, E. C., 2000, . *Experience with novices: The importance of graphical representations in supporting mental models.* Paper presented at the Proceedings of the 12th Workshop of the Psychology of Programming Interest Group, Corigliano, Calabro, Cosenza, Italy.

Ghefaili, A. (2003). Cognitive Apprenticeship, Technology, and the Contextualization of Learning Environments (PDF). *Journal of Educational Computing, Design& Online Learning, Vol. 4*, (Fall, 2003), 1-27.

Götschi, T., Sanders, I., & Galpin, V. (2003). *Mental models of recursion*. In *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education* (Reno, Navada, USA, February 19-23, 2003).SIGCSE'03.ACM,NewYork,NY,346-350

Greca, I. M., & Moreira, M. A. . (2002). Mental, physical, and mathematical models in the teaching and learning of physics. *Science Education, 86*(1), 106-121.

Hay, K. E., & Barab, S. A. (2001).Constructivism in practice: A comparison and contrast between apprenticeship and constructionist learning environments. *The Journal of The Learning Sciences, 10*(3), 281-322.

Huang, S., Cho, Y., & Lin, Y. (2006). *Implementation and Evaluation of Teaching an Introductory Software Engineering Course Framed in Cognitive Apprenticeship*.

Proceedings of the 2008 Conference on Future Play: Research, Play, Share, IEEE Computer Society, 477-484.

Järvelä, S. (1995). The cognitive apprenticeship model in a technologically rich learning environment: Interpreting the learning interaction. *Learning and Instruction, Volume 5*(3), 237-259.

Jih, H., J., & Reeves, T., R. (1992). Mental models: A research focus for interactive learning systems *Educational Technology Research and Development, 40*(3), 39-53.

Johnson- Laird, P. N. (1983). *Mental models*. Cambridge, UK: Cambridge University Press.

Lave, J., & Wenger, E. . (1991). *Situated learning: Legitimate peripheral participation*. Cambridge: CUP.

Ma, L., Ferguson, J., Roper, M., & Wood, M. (2007). *Investigating the viability of mental models held by novice programmers*. SIGCSE Bull., 39(1), 499-503.

Miquidade, A., A., & Chinama, S. (2003). Proposta de Currículo do Curso de Licenciatura em Informática de Gestão. *Instituto Superior Politécnico e Universitário ESGT*(1), pp. 1-74.

Norman, D. A. (Ed.). (1983). *Some observations on mental models*. Hillsdale, NJ: Lawrence Erlbaum.

Potosnak, K. (1989). Mental models: Helping users understand software. *IEEE Software, 6*(5), 85–88.

Redish, E. F. (1994). The implications of cognitive studies for teaching physics. *American Journal of Physics, 62*(6), 796-803.

Sanders, I., Galpin, V., & Gotschi, T. (2006). *Mental Models of Recursion Revisted.* Paper presented at the ITiCSE'06, Bologna, Italy.

Schulte, C., Magenheim, J., Niere, J., & Schäfer, W. . (2003). Thinking in Objects and their Collaboration: Introducing Object-Oriented Technology. *Computer Science Education, Vol. 13*(Issue 4,), pp. 269-288.

Seel, N., M., Al-Diban, S., & Blumschein, P. (2000). Mental Models & Instructional Planning. In J. M. S. a. T. M. A. (eds) (Ed.), *Integrated and Holistic Perspectives on Learning, Instruction and Technology* (pp. 129-158). Dordrecht, Netherlands: Kluwer Academic Publishers.

Shabo, A., Guzdial ,M., & Stasko,J. (1997). An apprenticeship-based multimedia courseware for computer graphics studies provided on the World Wide Web *Computer & Education, Vol. 29*(Issues 2-3), pp. 103-116

Sharad, S. (2007). Introducing Embedded Design Concepts to Freshmen and Sophomore Engineering Students with LEGO MINDSTORMS NXT. *IEEE International Conference on Microelectronic Systems Education, 2007, 3-4*(3-4), pp. 19 - 120.

Stelios, X., & Maya, S. (2005). *The hands-on activities of the programming microworld objectKarel*. Paper presented at the Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education, Caparica, Portugal.

Van Gorp, M., J., & Grisson, S. (2001). An Emperical Evaluation of Using Constructive Classroom Activities to Teach Introductory Programming. *Computer Science Education,*

Vol. 11(No. 3), pp. 247-260.

Vara, C., F., & Tan, P. (2008). *The game studies practicum: applying situated learning to teach professional practices*. Paper presented at the Conference Proceedings of the 2008 Conference on Future Play: Research, Play, Share, Toronto, Ontario Canada.

Westbrook, L. (2006). Mental Models: a theoretical overview and preliminary study. *Journal of Information Science, 32*(6), 563-579.

Zdeslav, H. (2002). *Identifying Students' Models of Sound Propagation.* Paper presented at the Proceedings of 2002 Physics Education Research Conference., Boise, Idaho.