A Conceptual Metaphor Based Model for Enhanced Understanding of Programming Concepts

Moape T. Gorgina
MoapeTG@tut.ac.za

Sunday O. Ojo
OjoSO@tut.ac.za

Department of Computer Science
Tshwane University of Technology
Pretoria, Private Bag, X680, South Africa

Manoj Lall
LallM@tut.ac.za
Department of Computer Science
Tshwane University of Technology
Pretoria, Private Bag, X680, South Africa

Abstract

Computer programming concepts are inherently abstract in form and intangible in essence, and so, to many a student, they are intuitively challenging to understand. This is a contributory factor to high students' failure rate in higher education introductory computing courses. Metaphors are known to be central to ordinary natural language semantics, as an enabler of conceptualization of one mental domain in terms of another. Studies in various contexts have shown that conceptual metaphors have assisted learners to develop better conceptual understanding by linking the known concepts to abstract programming concepts through a cognitive mapping process. In this paper we explore the use of metaphors in the teaching and learning of computer programming concepts to enhance understanding in introductory courses. The importance of using metaphors in the programming teaching process is emphasized and a metaphor based model for programming concepts teaching is proposed. The model is based on the Conceptual Metaphor Theory and Experiential Learning Theory, which guide the metaphoric cognitive process of using an existing known knowledge for illustrating new knowledge (programming concepts). Our direction of further works to operationalize the model into a pedagogical platform for teaching and learning of computer programming is charted.

Keywords: Programming, programming concepts, model, metaphor theory, experiential learning.

1. INTRODUCTION

"Technology and computers are very much at the core of our economy going forward, to be prepared for the demands of the 21st century—and to take advantage of its opportunities—it is essential that more of our students today learn basic computer programming skills, no matter what field of work they want to pursue", says Todd Park U.S. Chief Technology Officer. We are living in the world where technology is evolving on a daily basis and computer programming form part of the core fundamentals of developing these technologies. Computer programming is considered an important competence for the development of higher-order thinking in addition to problem solving skills and its horizontal integration throughout all educational levels and is considered worthwhile and attracts the attention of researchers (Fessakis, Gouli & Mavroudi, 2013)**.**

Computer programming concepts are inherently abstract in form and intangible in essence, and so, to many a student, they are intuitively challenging to understand. As a consequence, learners of computer programming are faced with many difficulties, including high cognitive load, in the process of learning (Yousoof, Sapiyan & Kamaluddin, 2007). This is a contributory factor to high students' failure rate in higher education introductory computing courses(Dann, Cooper, and Pausch 2006; Ali, 2009). For long, metaphors are known to be central to ordinary natural language semantics, as an enabler of conceptualization of one mental domain in terms of another (Lakoff & Johnson, 1980; Lakoff, 1993). In the field of computing, the pedagogical role of metaphors is well acknowledged (Umar & Hui, 2011; Hui & Umar, 2011; Forisek & Steinova, 2012; Bell, Duncan, Jarman & Newton, 2014), and in fact, the language of the field in general, and software development/programming in particular, is laced with metaphor (Colburn & Shute, 2008) Metaphorical use of shopping carts for complicated data structures in web computing, garbage collection in operating systems, and trees, stack and queues as data types in programing, are just but a few examples of such. This is towards reducing cognitive load in learning (Yousoof et al, 2007), and consequently, enhanced understanding of abstract conceptualizations that are characteristics of the field of computing in general, and programming concepts in particular.

Metaphor has been characterized as "a description of an object or event, real or imagined, using concepts that cannot be applied to the object or event in a conventional way" (Indurkhya, 1993). In the field of computing, abstraction is a key conceptual paradigm used to manage complexity through information hiding, and a metaphor is used as a vehicle to enable mental conceptualisation of abstractions to gain understanding of the underlying concept. Studies in various contexts have shown that conceptual metaphors have assisted learners to develop better understanding by linking the known concepts to abstract programming concepts through a cognitive mapping process. The pedagogical role of metaphor serving as such a vehicle has been widely recognized among computer science education researchers. However, the need to adopt a theory-driven approach to employing metaphor in the teaching and learning of programming concepts has not been adequately recognized. Metaphor used as a pedagogical vehicle goes beyond being just for naming abstraction, rather, it is a cognitive process that occurs when someone seeks understanding of one idea (the target domain) in terms of a different, already known idea (the source domain) (Hui, 2009). There is the need for a theory-driven approach to using metaphors in the teaching and learning of computer programming concepts to further enhance understanding in introductory courses.

The purpose of this paper is to report on an on-going study to investigate the role of metaphors in understanding Computer Science concepts in general with particular focus on computer programming concepts and present a metaphor based model for enhancing understanding of computer programming concepts, using Hui(2009)'s Self-Conceptual Metaphor Theory as a basis. A proposal for further works on the proposed model is also discussed. It is intended that the proposed model will provide a nucleus for creating a platform for developing pedagogical tools and techniques for enhancing teaching and learning effectiveness, towards improved student academic performance.

2. LITERATURE REVIEW

Metaphor and Experiential Learning

A Metaphor is a cognitive process that occurs when a subject seeks understanding of one idea (the target domain) in terms of a different, already known idea (the source domain) (Hui, 2009). An individual creates a conceptual mapping between the properties of the source and the target, thereby gaining new understanding about the target (Forišek & Steinová, 2012). Human conceptual system which is largely metaphoric, plays a central role in defining everyday realities, and so. humans do every day is very much a matter of metaphor (Lakoff & Johnson, 1980). The use of metaphors as a vehicle for learning unfamiliar concepts is widely acknowledged. There is general agreement that metaphors can be used not only to communicate the relevant properties of the new concept, but also to facilitate developing new conceptual structures, i.e. new mental models and new abstractions (Forišek & Steinová, 2012; Lakoff, 1993). The role of metaphor in understanding concepts in science(Kuhn, 1992) in general and in Computer Science in particular is widely recognized (Forišek & Steinová, 2012; Hui & Umar, 2011; Jiménez-Peris *et al.*, 1997). Studies have shown that metaphors have assisted learners to develop better conceptual understanding by linking the known to newly acquired abstract programming concepts (Blasco-Arcas *et al.*, 2013; Sorva, Karavirta & Malmi, 2013)

Learning from experience is one of the most fundamental and natural means of learning available to everyone, it need not be expensive, nor does it require vast amounts of technological hardware and software to support the learning process, instead, in the majority of cases, all it requires is the opportunity to reflect and think, either alone or in the company of other people (Beard & Wilson, 2006). Experiential learning involves learner centred opportunities predicated on the idea that individuals learn best by experience and it occurs within an immediate and relevant setting; promotes the acquisition and application of knowledge, skills, and feelings (Hakim *et al.*, 2014). According to Kolb (1984) experience plays an essential role in learning, if used and applied properly, it has a great potential of minimizing the learning curve in education.

Given that metaphors are intrinsically contextual (Forišek & Steinová; 2012), a combination of a metaphor object with learner's experiential knowledge of the object is necessary in any metaphoric model to be used as a pedagogical vehicle for enhancing programming concepts understanding. This view is an element that underpins our proposed model.

Metaphor Use in Programming Learning

Metaphor is rather a process than an entity, a process whereby an understanding of one familiar entity or existing knowledge is used to illustrate and explain unfamiliar entity or new knowledge. According to the Oxford Dictionary, metaphor is a figure of speech in which a word or phrase is applied to an object or action to which it is not literally applicable, it continues to define it as a thing regarded as representative or symbolic of something else. A conceptual metaphor is a cognitive process that occurs when a subject seeks understanding of one idea (the target domain) in terms of a different, already known idea (the source domain) (Forišek & Steinová, 2012). As defined by Lakoff (1993) the target is the subject to which attributes are assigned, unfamiliar concept and the source is the subject from which attributes are borrowed, familiar concept, that is called to describe the target. There is general agreement that metaphors can be used not only to communicate the relevant properties of the new concept, but also to facilitate developing new conceptual structures (new mental models and new abstractions) (Hidalgo-Céspedes, Marín-Raventós & Lara-Villagrán, 2014; Hui & Umar, 2011; Rieber & Noah, 2008; Sanford *et al.*, 2014). Studies have shown that metaphors have assisted learners to develop better conceptual understanding by linking the known to newly acquired abstract programming concepts (Jiménez-Peris *et al.*, 1997; Kuljis & P Baldwin, 2000; Umar & Tie, 2011; Waguespack Jr, 1989). Hidalgo-Céspedes *et al.* (2014) conducted an observation that when learning computer programming, students must mentally construct abstract concepts like pointers and threads by associating them with other concepts acquired in in their life experience. The authors constructed a video game as a pedagogical tool for students' better understanding of programming concepts.The video game consists of metaphors used to represent abstract programming concepts and colloquial concepts that resemble abstract ones. When students follow gameplay rules, they will automatically learn abstract programming rules, they authors opined. However, no theoretical basis was provided for their model. Weather this video game had a significant impact on the understanding of programming concepts by students or not it is unknown as the results are not documented in the paper.

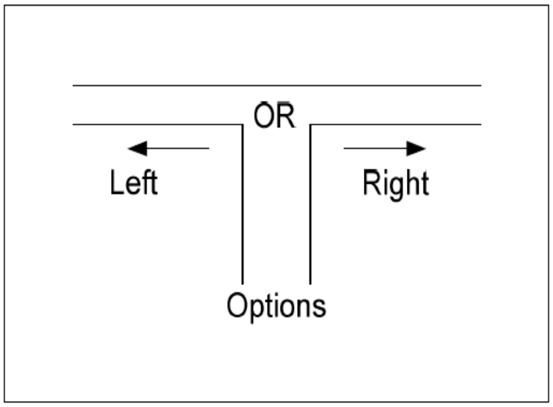

Figure 1: T-junction road routes metaphor (HUI H. T. (2009))

Video game increase learning engagement and facilitates an environment of experiential learning however, it requires more time and may not be consistent with learning objectives. HUI H. T. (2009) discusses the essential difficulties uncounted in teaching and learning programming. He further instigates a self-conceptual metaphor approach that can be used to improve the understanding of programming concepts and strengthen student's coding practices. Figure 1 and Figure 2 points out examples of conceptual metaphor used to demonstrate programming concepts by the use of conceptual metaphor.

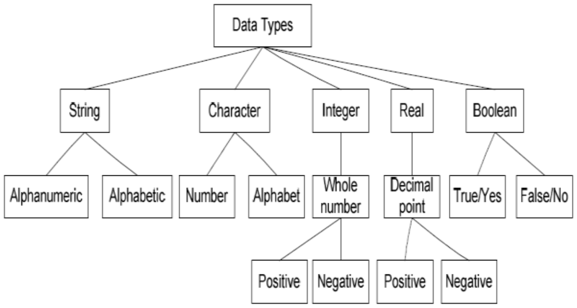

Figure 2: Tree metaphor (HUI H. T. (2009))

The T-junction road metaphor is used to illustrate an if-statement in programming. T-Junction is a two-sided option concept where one option is required, either taking the right turn or a left. The IF, ELSE statement could be interpreted as a choice to be chosen from a list of 2 options which yields one of two possibilities: true (yes) or false (no). The "IF statement" should be understood better when explained by making use of the T-junction road routes metaphor. In programming, there are different types of data types which are essential for students to understand. To illustrate this concept, HUI H. T. (2009) employed the tree metaphor. A tree consists of trunk, branches, stems and in programming data types consists of integers, string, Boolean and characters. Understanding of data types will be enhanced by use of a tree metaphor. It is essential that the source subject is represented as a real world system to enable students to apply as a reference for linking the existing ideas to the newly introduced programming concepts (Hui & Umar, 2011). Ali (2009) proposed a Conceptual Model to simplify learning to program in introductory Programming courses. The model was developed alongside the System Development Life Circle (SDLC) and consists of three phases, namely; Planning phase, Coding phase and Output phase. He adapted the use of metaphor on output phase of the model where visual metaphors are used to engage the programmer; however, no underlying theory was presented. The objective of the model is to simplify the steps involved in programming from algorithm writing to output execution. Waguespack Jr (1989) developed a system of metaphor where he used shapes as metaphor to illustrate different data types in programming. The following figure shows shapes used as metaphors.

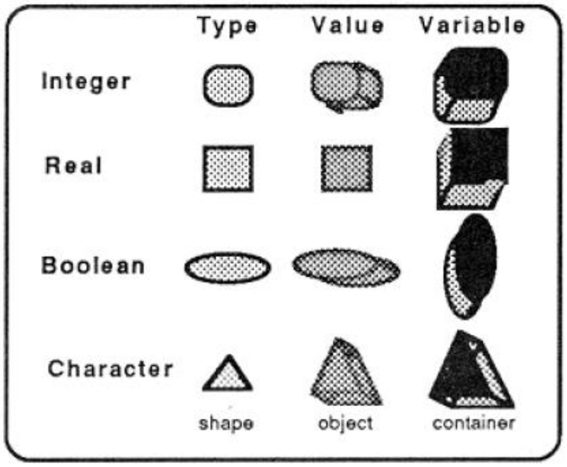

Figure 3: Shapes metaphor to explain data types (Waguespack Jr, 1989).

Integers, real numbers, Boolean data types and characters are basic data types in every programming language. A variable of a particular type is a container with the corresponding particular shape, data values of the same type as the variable can be placed in that variable - the appropriately shaped object can fit in a container with the same shape and incompatible types resist such assignment because "you can't put a round peg in a square hole (Waguespack Jr, 1989). Hui and Umar (2011) conducted a survey to investigate if the combination of metaphor and pair activity helps enhance programming performance and recall of students with different Self-Regulated Learning. The authors conducted Motivated Strategies for learning Questionnaires (MSLQ) to identify students with high self-regulated learning (SRL) and students with low Self-regulated learning. SRL is defined as a process where by students set goals for their learning. To administer their experimental study, students were assembled in pairs according to their SRL (one with high and low) and assigned to two groups. One group received Metaphor Pair Programming (MPP) and the other group will receive Pair Programming (PP) when solving programming problems. Based on the performance of students, it was discovered that metaphor and pair programming assist learners to understand programming concepts better and students with high SRL performed better that students with low SRL. The following Figure 4 is the medical capsule metaphor used to represent a class and Figure 5 is the library metaphor used to represent the class concept.

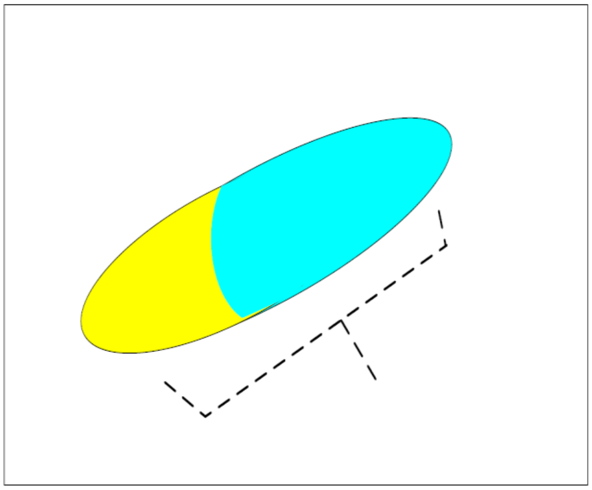

Figure 4: Medical Capsule representing Classes(Hui & Umar, 2011)

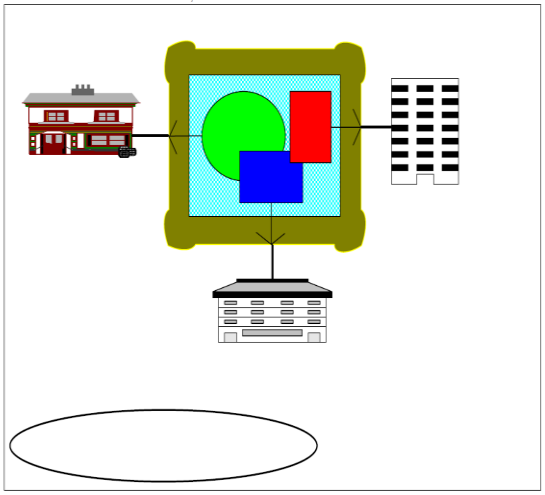

Figure 5: Library representing a Class Concept(Hui & Umar, 2011).

A common observation in the studies referenced above is that the need for a theory-driven approach to evolving metaphoric model for enhancing programming concepts understanding is not recognized. As a result, they are lacking in a systematic approach to selecting suitably metaphor object and ascertaining ontological compatibility of the selected metaphor object with the target programming concept. Consequently, the proposed metaphors could be limited with regards to their fitness for purpose. This is a gap addressed in our theory-driven approach.

3. OUR APPROACH

Theoretical Framework

The proposed model in this study is adapted from the Self-Conceptual Metaphor by Tie Hui Hui (2009). Hui (2009) used the Self-Conceptual Metaphor Theory as a basis for instructional method to teach programming. The model uses metaphor to enable students to engage in linking the known to the unknown. Figure 6 shows the fundamental model of self-conceptual metaphor as proposed by Hui. Existing knowledge refers to knowledge already known to an individual. The enhancements to information systems education is identified or repeated here. DO NOT repeat the abstract or portions of it.

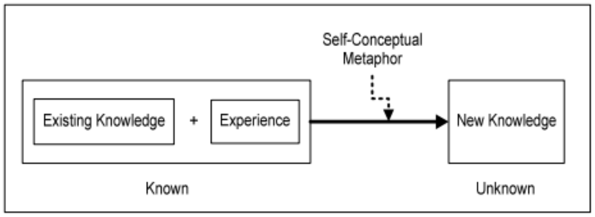

Figure 6: The Fundamental Model of Self Conceptual Metaphor. (Hui 2009)

Self-Conceptual metaphor is a cognitive process that occurs when an individual seeks understanding of the new knowledge (the target domain) in terms of a different, already known idea (existing knowledge-source domain) based on the individual's experience with that existing knowledge. According to Hui (2009), relating the model to student learning, the self-conceptual concept is first engaged when a student discovers new knowledge, and then the existing knowledge or idea is carefully analysed in order to set connectivity between these two elements (unknown to known). Finally, the link between the known and unknown is established and a better understanding of the new concepts will be developed, he opines. He further posits that the knowledge based on previous experiences of the students will influence the outcome of the writing programs and their individual understanding of unknown concepts. He concurred with Kolb (1984) that prior experiences influence how learners learn. The model concludes that the fundamental of self-conceptual metaphors are the metaphorical ideas or terms used by the individual students during their programming activities are very natural and could vary from other students. This is because of the individual’s understanding and past experiences. (Forišek & Steinová, 2012) later corroborated this view point that a metaphor can serve as a vehicle for understanding a concept only by its experiential basis.

Proposed Model

The proposed model presented in this paper is based on the Self Conceptual Metaphor theory by Hui (2009), Kolb (1984) and Kolb & Kolb (2012)’s Experiential Learning Theory. Figure 7 shows our proposed model as adapted from Hui (2009).

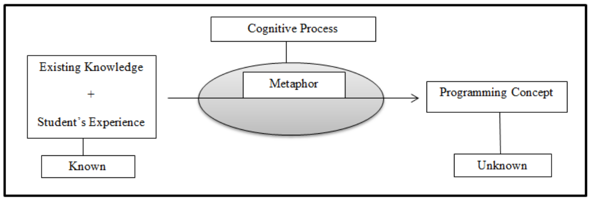

Figure 7: Proposed Metaphor Based Model for understanding programming concepts (adapted from Hui (2009)).

According to Kolb (1984) and Kolb & Kolb (2012), learning is a process whereby new knowledge is acquired and builds upon existing knowledge. Experimental Learning Theory states that learning is a process which is created through the transformation of experience. In the process of learning, an individual is called to exercise back and forth between opposing modes of reflection and action (Robinson, 2012). In the model, Existing Knowledge is what is already known to the student. Student’s experience is an occurrence or involvement with a specific entity or a situation. The existing knowledge together with the experience form what is known to the student. Metaphor is a cognitive process by which an understanding of one thing is used to explain another. Cognitive process is whereby students establish a relationship between their existing knowledge and new knowledge. Programming concepts (unknown) will be taught to learners by making use of the students experience and their existing knowledge (known). By relating new knowledge with the existing knowledge, a better understanding is developed. Programming concepts are constructs used in programming.

To examine appropriates and ontological mappings between source object and target object will adopt the Contemporary Theory of Metaphor by George Lakoff. The principle of understanding the domain of a source object in terms of the domain of a target will be used. In selecting our source object will follow the same principle (ontological mappings) by Lakoff to measure the appropriateness and the compatibility between source objects and target objects (Programming Concepts). The ontological mappings between these objects will also be established as proof of suitability. Forišek and Steinová (2012) define ontological mappings as a portray from a source domain to a target domain, with the source domain defined as the familiar, real-world concept that maps to the target domain, defined as the concept that is being taught.

Other methods that have introduced to alter the difficulty of understanding programming concepts are videos. These videos are self-learning videos which requires separate time and attention which students don’t make time for. The video’s consists of slides and a person talking in the background explaining contents of the slides which is more or less the same ineffective method used during lectures. Except for the fact that videos are time consuming, they are less accommodative on the eye.

The proposed model should be more effective because it utilizes concepts that are familiar with student. The metaphor objects (Source Objects) used will be drawn from students’ common experience. These source objects will be sourced from University culture. Learners will relate more to the source objects and relationship between old and new concepts will be better established. The goal of the proposed model is not only to enhance understanding but to increase student’s engagement in class.

4. MODEL OPERATIONALIZATION

As shown in figures 8, 9 and 10, the Automated Teller Machine (ATM) is used as a Metaphor Object to illustrate some programming concepts of Data Processing and Control Structures. Figure 8 shows the operationalization of our model in figure 7. It depicts ATM Knowledge + Student’s Experience as an instantiation of Existing Knowledge + Experience (in figure 7), being linked with the Programming Concepts (Unknown Knowledge). The mapping is an ontological mapping from the ATM concept to the Programming Concept through a metaphoric cognitive process.

The scenario represented as figure 9 with ATM as a Metaphor Object and Data Processing concept as the unknown knowledge, can be described thus: To withdraw cash from the ATM, a procedure has to be followed, you cannot start by entering pin code without inserting the card first, and this procedure is used to illustrate the sequence control structure in programming. Before the ATM can provide cash, it first checks if you have sufficient funds to be withdrawn, based on the results, you will be provided with cash or "you have an insufficient funds" statement as output.

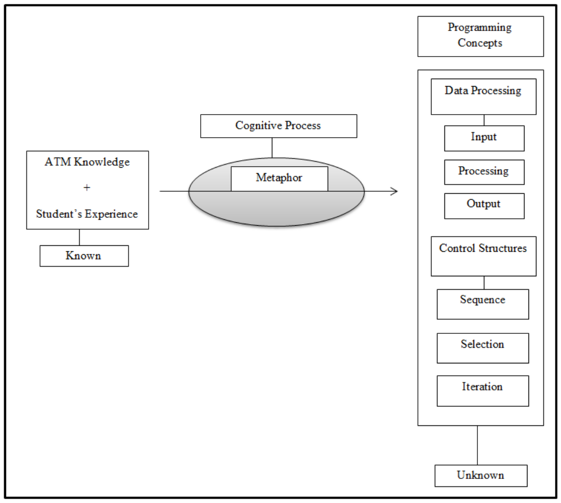

Figure 8: Model Operationalization

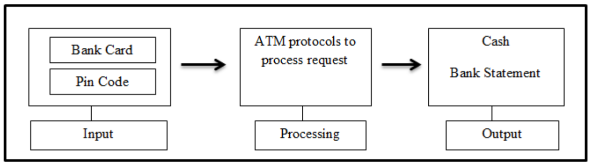

Figure 9: ATM Metaphor to illustrate Data Processing Concept

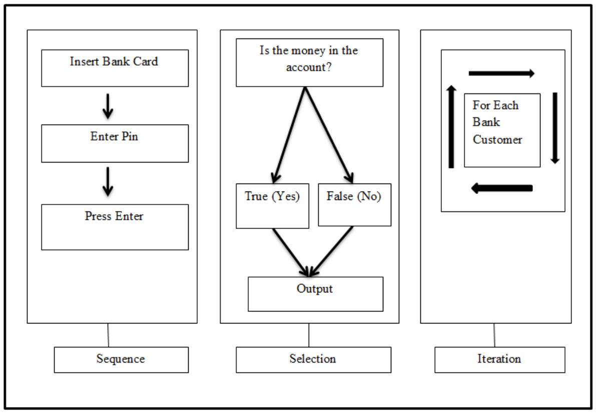

Figure 10: ATM Metaphor to illustrate Control Structure Concepts

This metaphor is used to illustrate the Selection control structure. For each customer, the same procedure is followed. This is used to illustrate Iteration. Figure 10 shows the relationship between an ATM (Metaphor Object) and Programming Control structures.

5. SUMMARY AND FUTURE WORK

In this paper, we first explored the computer programming in general, its complexity and requirements. The paper reviewed two most important factors that can be used to alter the difficulty and complexity of programming which is Metaphor and Experience. We have proposed a metaphor based model for enhancing understanding of programming concepts, using Hui (2009)'s Fundamental Self-Conceptual Model as a theoretical basis. We further explained the pattern (ontological mapping) that will be used to ensure appropriateness between source objects and target objects.

This being a work-in-progress, the next step is probe the University culture, carefully select metaphor objects that will utilized to explain programming concepts. We intend to develop a prototype implementation of the model, operationalized it with more metaphoric ideas for different programming concepts, and evaluate the prototype using expert judgement of computer programming lecturers in a university environment.

It intended that the implementation realization of the proposed model will provide a platform for developing pedagogical tools and techniques for computer programming conceptual understanding and teaching and learning effectiveness, towards improved student academic performance.

6. ACKNOWLEDGEMENTS

We would like to acknowledge the Tshwane University of Technology for providing a Masters Student bursary for the on-going project.

7. REFERENCES

Ali, A. 2009. A conceptual model for learning to program in introductory programming courses. *Issues in Informing Science and Information Technology,* 6(1):517-529.

Beard, C. M. & Wilson, J. P. 2006. *Experiential learning: A best practice handbook for educators and trainers*. Kogan Page Publishers.

Bell T, Duncan C, Jarmans S, & Newton H (2014). Presenting Computer Science Concepts to High School Students. Olympiads in Informatics Vol 8, 3-19

Blasco-Arcas, L., Buil, I., Hernandez-Ortega, B. & Sese, F. J. 2013. Using clickers in class. The role of interactivity, active collaborative learning and engagement in learning performance. *Computers & Education,* 62:102-110.

Fessakis, G., Gouli, E. & Mavroudi, E. 2013. Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study. *Computers & Education,* 63:87-97.

Forisek, M. & Steinova, M. 2012. Metaphors and analogies for teaching algorithms. Paper presented at the *Proceedings of the 43rd ACM technical symposium on Computer Science Education*.

Hakim, E. W., DSCPT, M., MARILYN MOFFAT, P., ELAINE BECKER, P., BELL, K. A., MANAL, T. J., et al. 2014. Application of educational theory and evidence in support of an integrated model of clinical education. *Journal of Physical Therapy Education,* 28:13.

Hidalgo-Cespedes, J., Marin-Raventos, G. & Lara-Villagran, V. 2014. Playing with metaphors: a methodology to design video games for learning abstract programming concepts. Paper presented at the *Proceedings of the 2014 conference on Innovation & technology in computer science education*.

Hui, T. H. 2009. Self-Conceptual Metaphor: An Alternative Instructional Strategy in Delivering the Fundamental Programming Concepts to Novices. Retrieved from http://www.segi.edu.my/onlinereview/chapters/vol2_chap3.pdf

Hui, T. H. & Umar, I. N. 2011. Does a Combination of Metaphor and Pairing Activity Help Programming Performance of Students with Different Self-Regulated Learning Level? *Turkish Online Journal of Educational Technology-TOJET,* 10(4):121-129.

Indrikhya B. (1992), Metaphor and cognition, Kluwer Academic Publishers, Dordrecht, The Netherlands.

Kolb, D. 1984. Kolb, DA (1984): Experiential learning: experience as the.

Kuljis, J. & P Baldwin, L. 2000. Visualisation techniques for learning and teaching programming. *CIT. Journal of computing and information technology,* 8(4):285-291.

Kuhn T. (1992) Metaphor in science. In: Ortony A.(Ed.), Metaphor and Thought, Cambridge University Press, Cambridge, 1993. pp. 533–542.

Lakoff, G. 1993. The contemporary theory of metaphor. *Metaphor and thought,* 2:202-251.

Lakoff, G. & Johnson, M. 1980. Conceptual metaphor in everyday language. *The journal of Philosophy*:453-486.

Rieber, L. P. & Noah, D. 2008. Games, simulations, and visual metaphors in education: antagonism between enjoyment and learning. *Educational Media International,* 45(2):77-92.

Robinson, J. S. 2012. Aligning Kolb's experiential learning theory with a comprehensive agricultural education model. *Aligning Kolb's Experiential Learning Theory with a Comprehensive Agricultural Education Model,* 53(4):1-16.

Sanford, J. P., Tietz, A., Farooq, S., Guyer, S. & Shapiro, R. B. 2014. Metaphors we teach by. Paper presented at the *Proceedings of the 45th ACM technical symposium on Computer science education*.

Sorva, J., Karavirta, V. & Malmi, L. 2013. A review of generic program visualization systems for introductory programming education. *ACM Transactions on Computing Education (TOCE),* 13(4):15.

Umar, I. & Hui, T. H. 2011. Metaphors and pairing activity for comprehending programming concepts in computing education*.* Paper presented at the *Global Learn*.

Waguepack Jr, L. J. 1989. *Visual metaphors for teaching programming concepts*. (21). ACM.

Yousoof M, Sapiyan M & Kamaluddin K (2007). Reducing Cognitive Load in Learning Computer Programming. International Journal of Computer, Electrical and Information Engineering 1(12), 4087-4090