I.J. Modern Education and Computer Science, **2014, 9, 15-21**
Published Online September 2014 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2014.09.03

A Methodology for Teaching Computer Programming: first year students' perspective

Bassey Isong
Department of Computer Science & Information Systems, Thohoyandou, South Africa
bassey.isong@univen.ac.za

Abstract—The teaching of computer programming is one of the greatest challenges that have remained for years in Computer Science Education. A particular case is computer programming course for the beginners. While the traditional objectivist lecture-based approaches do not actively engage students to achieve their learning outcome, we believe that integrating some cutting-edge processes and practices like agile method into the teaching approaches will be leverage. Agile software development has gained widespread popularity and acceptance in the software industry and integrating the ideas into teaching will be constructive. In the educational system, while the positive impact of agile principles has been felt on students' projects, none has been experienced on the teaching aspect. Therefore, this paper proposes the use of agile process in the teaching of first year programming courses. The goal is to help the beginners develop their programming skills, proffer a teaching technology that maximizes students' chances of engagement, improve teaching as teachers reflects on what they are teaching and what the students are learning. Additionally, beginners will be able to operate the computer, program, and improve their programming skills through active team collaboration as well as managing large classes effectively by the teacher.

Index Terms—Programming, Agile process, Pair programming, Students, Teaching.

I. Introduction

In higher institutions of learning in most developing countries today, teachers are facing profound teaching and learning challenges. These challenges have been attributed largely to factors such as increasing student's diversity, under-preparedness, language competencies, large classes, poor class attendance, poor teaching methodologies, and so on. There are all direct effects of what Northedge [1] described as

"...continuing pressure towards greater relevance and widening of participation".

Taking teaching methodology into consideration, in years back when university classes were confined to selected high quality students, the traditional lecture alongside tutorial was believed to be working well enough. But today, due to the exponential growth of diversity of students' population, many students seem not to be coping with the system in which majority of the students have been described in one way or the other as "not being university material" due to under-preparedness [2]. This is an epitome of the case of beginners offering computer programming (CP) courses in higher education (HE) today. While this is true in several universities in developing countries, one of the important goals of computer science (CS) teachers is to assist these students to achieve their learning outcome [1]. In order not to undermine the objectives of HE, this teaching challenges should not to be seen as an impediment to teaching those students but it should be geared towards changing their usual way of learning and perception. Given the critical context it is important that a teaching technology that maximizes the chances of engaging students' learning processes should be used [2].

There are many different methods that exist today which have been developed and utilized in the context of teaching and learning. One of such approaches is the information and communication technology (ICT) based approach. The technologies have been commended for its improved effectiveness and efficiency in the educational process worldwide [3]. The rationale for the incorporation of ICT-based approaches into the educational process is due to the rapid global technological advancement being witnessed in recent years. In this case, the educational methodologies and customs have to be changed as well in order not to remain obsolete. In addition, being confronted with increasing diversity, teaching despite the benefits of the ICT-based tools is becoming a central focus and even most challenging tasks today [5]. Most students are not engaged and the system is not working as designed. Therefore, a new technology-rich approach is required to cope with the diverse and dynamic needs of students [4] [5]. The achievement of ICT in the educational process is that it has the capacity to improve student's learning outside the actual boundaries of the classroom. This is evident as students today can study and learn new things using their mobile phones, the Internet, computers, and so on. In a nutshell, ICT in education has been established for improving memory retention, increasing motivation and deepens understanding [6]. Furthermore, it also promotes collaborative learning, group problem solving activities, role playing and so on [7].

In most universities today, difficulties are still being experienced even with the use of ICT-based tools [6]. In particular, the teaching of introductory CP course for beginners has been considered challenging. The teaching approach needs to be revamped technologically to further develop it. The essence is that the current teaching approach of CP courses has always assumed the traditional objectivist lecture-based and the instructional design for lecture delivery have not changed either even as computer technology and curriculum have undergone changes [27]. Consequently, students are not engaged which makes it difficult for the students to understand basic programming concepts. This paper therefore, argues that the traditional teaching approach coupled with fixed instructional design is the main cause of why learning and teaching of CP still remains difficult. Our position in this paper is that, in order to address these challenges, it is imperative that CS teachers have to improve their teaching approaches with not just using ICT facilitated learning environments, but the incorporation of advanced ICT practices and processes into the teaching activities. In this case, this paper proposes a new approach of teaching first year CP course using the practices of agile software development methodology. The objective is to promote active students' participation and engagement as well as helping students grasps difficult programming concepts as programming skills is considered difficult to acquire.

The rest of the paper is organized as follows: Section II is the study background, Section III introduces agile software development process, Section IV is the challenges that affect the teaching and learning of programming, Section V is the proposed methodology, Section VI is the study discussions and Section VII is the conclusion.

II. Background Information

In the CS discipline in particular and the engineering discipline in general, programming constitutes one of the pillars of computing. It is considered a useful skill which is one of the nucleus competences expected of every of CS graduate [8]. CP is described by Deek & McHugh [9] as a problem solving process of formulating, planning and designing the solution, translating, testing, and delivery. For a person to be proficient in programming, essential skills such as learning the language, composing new programs, debugging, understanding, reusing and integrating existing programs are essential[9][10][22]. Unfortunately, CP has been labeled difficult involving a huge cognitive activity that requires students to perform exhaustive and extended practices in order to develop the needed essential programming skills and gain the needed experience [8][10][23][24]. Generally, the teaching and learning of programming poses one of the greatest challenges in the area of CS Education [25][26].

Due to the nature of CP courses in the universities education these days, students, especially, beginners students can easily lose enthusiasm and interests in learning CP. In addition, the continuous failures they experience during own practices can even worsen the situation [8]. Consequently, students' interest in the CS discipline could be declining as recently confirmed. For instance, a study by Foster [11], shows that students who indicated their desire to study CS declined by more than 60 percent between the fall of 2000 and 2004. Moreover, for some years now, there has been a dramatic change in the educational framework alongside computer technology for teaching CP and yet the instructional design for delivering such lectures still remain fixed [27]. Rolandsson [27] attributed this challenge to the rationale behind why the difficulties in learning and teaching of CP still exist. Though programming is difficult despite the use of ICT in teaching and learning, we argue that one of the several reasons why these problems continue to exist in our educational system is the teaching approach used in delivering CP to the students.

First year CS students are mostly students that have just left the secondary schools where CP is neither taught nor computer used. Several number of the students have never seen, use or touch the computer since their life time. Consequently, this poses a serious challenge to the teaching and learning activities. The challenge is exacerbated by the traditional objectivist lecture-based approach that is used in teaching the course. By using the traditional objectivist lecture-based approach, the teacher is the primary source of knowledge and learning is seen as an information transmission that originates from the teacher's knowledge to the students, driven by direct lecture [12]. Lectures take place only in classrooms and the computer laboratories are rarely used. Thus, with the unprepared nature of these students, lecturing programming in the classroom alone is challenging and sometimes it appears as if

"Russian language is used in teaching English language".

As a result, most students found it very difficult to use the computer or understand the function of a code and how they are written. Furthermore, students are not engaged and several of them are left non-participating as well as ill-motivated. According to Northedge [1] which says:

"….if we think of teaching as transmitting knowledge in the form of information, we seem to be faced with a conundrum".

Given the critical context it is imperative that sociocultural theories of learning which proffers a good choice of dealing with challenges of this nature should be adopted. With this approach, teaching is view as enabling participation in knowing [13]. In this case, students will be able to gain the required knowledge or skills in their learning process as comfortable atmosphere that enable them to actively engage and participate freely are created. This provides a clear motivation for the need to improve the teaching and learning of CP. The essence is that these students are the future of software development organizations and if they are not brought up in the right direction in their first year, it will have a negative impact on them and their career. Hence, the right step have to be taken in the right direction to ensure that they are well equip with the advanced technical savvy and core competencies expected of CS graduates.

In this study, we have proposed an approach for teaching the first year CP courses through the application of cutting-edge practices and processes of agile software development methodology. We believe that incorporating the practices of agile methodology into teaching will help shape the way these students learn and achieve their learning outcome. Accordingly, an approach that helps teachers improve their teaching will definitely help teachers reflect on what they are teaching as well as what their students are learning.

III. Agile Software Development Process

Agile software development process was founded in the 1990s by group of expert as a software development methodology involving several lightweight methods as part of the efforts against the traditional software processes or the "heavyweight" methods [14]. This was as a result of the rigid, slow, and inconsistent nature of software development exhibited by the "heavyweight" methods, waterfall model in particular [14][15]. The lightweight methodologies are designed to develop software in short iterations, increase productivity, involve customers throughout development period, and develop high quality software within time and budget [16]. Based on agile mode of operations, there is effective communication, project flexibility, increased customer satisfaction and changes are welcomed anytime during development [14].

Agile methods operate with what they called agile manifesto which was articulated in 2001 as a declaration of agile practices and principles [16][17]. The agile manifesto is as follows:

1) Individuals and interactions over processes and tools
2) Working software over comprehensive documentation
3) Customer collaboration over contract negotiation
4) Responding to change over following a plan

For more information on agile practices and principles, refer to [16]. With the articulation of agile manifesto, agile methods and principles have since gained importance and became extensively accepted in the software industry. It has now become a de factor standard which is evident in several software development companies that practices the agile methods nowadays. Given the benefits derived from agile practices, it is without dispute that the ideas embedded in agile have to be incorporated into teaching and learning in higher education in order to help students achieve their learning outcome in terms of both technical and social skills.

In the academic world today, agile principles are taught basically in lectures, academic conferences and several researches in the literature have reported the successful application of agile methods in student's projects or practical course. In particular, the uses of agile practices like pair programming in students' software development projects have been commendable [4][18][19]. With pair programming, all codes are programmed in pairs and it involves moving developers between pairs to ensure the participation of everyone in all development tasks. The benefits have been the promotion of collaborative learning and peer evaluation of each other's programming activities [18][21]. Nonetheless, to really integrate and adopt the agile values, theoretical lectures or agile principle in students' programming projects is not enough especially where students' diversity has become a serious challenge. CS teachers have to apply agile manifesto itself, blended with the practice of pair programming in their teaching. A particular case which needs urgent attention is the teaching of first year students' CP courses. This study argue that if agile practices are incorporated into teaching approach, it will go a long way to improving students' understanding of programming, promote active participation, collaboration and engagement.

IV. Challenges of Teaching Programming

Several challenges affect the teaching and learning of introductory CP course in most institutions of higher learning. These challenges are known to have some serious impact on both the teachers and students. These challenges include students' under-preparedness, lack of experienced CP teachers, lack of infrastructures such as well-equipped computer laboratory, lack of resources, and so on. While some of these challenges are easily dealt with by HE in developed countries, the case in most developing countries, especially the rural universities is different. Ideally, challenges that are specific to first year CS students which have been investigated are as follows:

A. Computer Literacy

Computer literacy poses one of the problems confronting the teaching and learning of CP in most universities. In the university system, aside some of the students who started their first year after they have undergone a one year remedial programme, it is believed that majority of other students in their first year are from the secondary schools where CP was never offered. In this case, they have not seen, touch or even operate computers before. Thus, they are only touching computers for the first time in their programming class. This however, makes it difficult for the students to operate the computer, understand programming concept and structures, syntax and semantics of the language, the programming environment and actively participate in CP classes.

B. Lack of Access to Computers and other Resources

This is another issue that affects teaching and learning of CP courses rooted from the poor socio-economic background of the undergraduate students. Financial constraints pose a great situation that indirectly affects teaching and learning of CP. In one way, due to the cost of securing accommodation within the university environment, most students are forced to live at home either with their parents or places nearby. Consequently, since they don't have personal computers or laptops of their own, the departmental laboratory is the only place where computers can be used for their practices or projects. Unfortunately, these students can only access the laboratory during the day since they live off campus and as a result, they cannot participate actively the class.

Additionally, majority of the students have no cash or means to purchase required CP text books. Only few students have bursaries while others that cannot afford text books on their own are forced to use library where only one or two copies of such books exists. As a result, they can't get the required resources to assist them in learning programming effectively.

C. *Language Competencies*

Language competencies are another serious challenge to teaching and learning in general and not programming only. Due to the foreign nature of English language, most of the students have poor English background and cannot communicate or write fluently with the language. Consequently, teachers have to choose words when teaching or try to mimic their mother tongue before the students can understand. Thus, students find it very hard to participate or engage actively in the classroom as they cannot communicate effectively either in writing or speaking.

D. *Lack of Experienced Programming Teachers*

Factor that immensely affects the teaching and learning of CP which is not from the students' side is sometimes due to scarcity of programming teachers or experienced teachers. Though, programming has been deemed difficult [23][24], sometimes this is not always true. It could be due to lack of experienced programming teachers or poor teaching methodology teachers used in delivering the lectures that does not actively engaged or encourage students' participation. Consequently, this affects the students' learning process. In this case, if a teaching methodology is found not to be productive, the students would not be productive as well unless other viable approaches are introduced.

These are some of the factors that contribute to the challenges being faced today by teaching and learning of CP courses in the classroom in most rural universities.

V. PROPOSED METHODOLOGY FOR TEACHING COMPUTER PROGRAMMING

In this study, the idea behind the proposal for a new teaching approach for CP is not based on the fact that the traditional objectivist lecture-based approach is completely bad. But rather, a teaching approach is needed that increases the changes of engaging students' learning processes, help teachers to reflect on their teaching and the students' learning. The idea is important in the light of students' growing under-preparedness, decline in CS programs enrolment, increasing numbers of unproductive CS graduates and so on. Therefore, using agile practices in the teaching of the beginner's CP courses will create a positive impact on the learning outcome of the students. Based on the agile manifesto and the pair programming practices, the sections that follow provides some descriptions of the proposed teaching methodology and the programming class set-up. This indicates that to engage the students in the CP class, the teaching methodology should assume those forms in its operations.

A. *Requirements*

For this proposal to be effective and successful, there are several requirements that needs to be satisfied. The requirements are as follows:

i. There should be available and well-furnished computer-equipped classrooms which can accommodate at least fifty (50) or 25 pairs of students offering introductory CP course, taking pair programming settings into account.
ii. All programming classes should not be held in ordinary classrooms rather than *computer-equipped classroom* or computer laboratories to avoid separate practical scheduling.
iii. Lecture notes and programming materials should be given to students before the commencement of the course.
iv. The teacher should be ready to actively maintain close contact with the students and not only teaching. In this case, he will be able to reflect on what he is teaching and what the students are learning.
v. Introductory CP classes should be scheduled to last for at least two hours per class.

These constitute the requirements that must be taking into consideration for the successful implementation of this proposal.

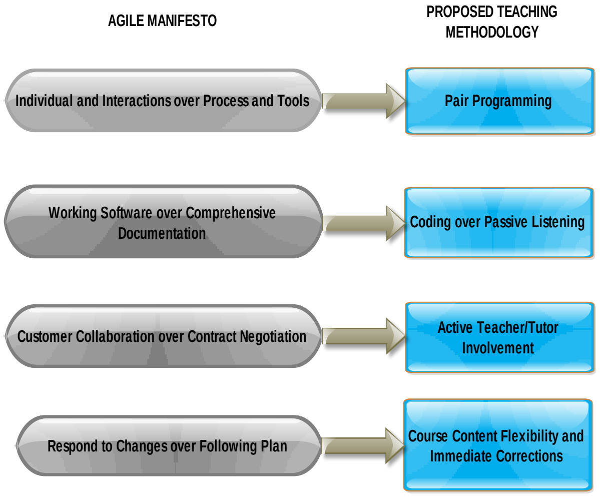

AGILE MANIFESTO
PROPOSED TEACHING METHODOLOGY
Individual and Interactions over Process and Tools
Pair Programming
Working Software over Comprehensive Documentation
Coding over Passive Listening
Customer Collaboration over Contract Negotiation
Active Teacher/Tutor Involvement
Respond to Changes over Following Plan
Course Content Flexibility and Immediate Corrections

Fig. 1. Agile manifesto and programming teaching alignment

B. *Agile Manifesto Alignment*

Considering the values implanted in agile manifesto, what is required in this study is that the teaching methodology for teaching CP to beginners in their first year should align to the principles of agile. This is captured in Fig. 1. The rationale is that aligning CP teaching in this manner would boost the motivation on the part of the students, get them engaged and participate actively. The approaches are discussed as follows:

1) Pair Programming: Here, the practice of pair programming and one integrated development environments (IDE) for programming is recommended to achieve the first agile norm which is "*individuals and interactions over processes and tools*". In this practice, two students should be assigned to one permanent computer in the laboratory to be used in every CP class. (see Fig. 2) While one student actively implements the program, the other should continuously observe the code with a view to identify errors and provide more ideas to achieve a better solution to their task as the teacher lectures. It is important that this practice is performed interchangeably among the students in order to maintain a rightful balance and avoid one student being passive. Additionally, the technique will help the students to develop strong trust on each other through continuous interactions will be improved. To master programming skills, it is adviced that students should not be allowed to use different tools for their development other than the recommended IDE by the teacher.

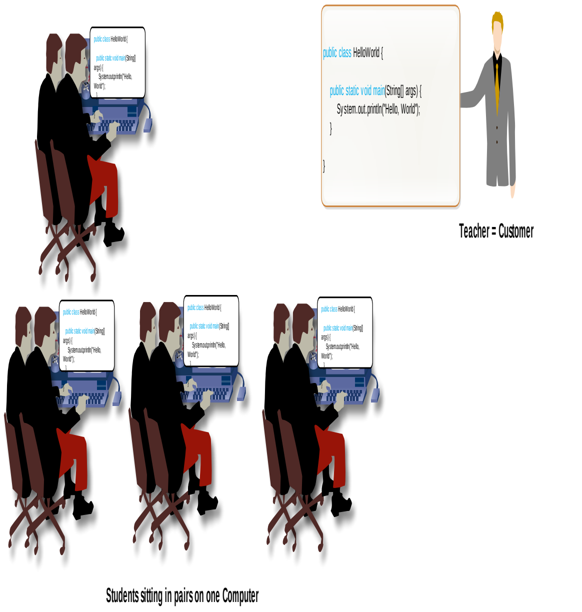
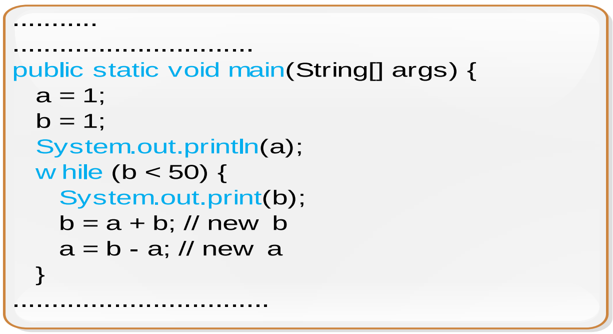

public class HelloWorld {
public static void main(String[] args) {
System.out.println("Hello, World");
}
}
Teacher = Customer
Students sitting in pairs on one Computer

Fig. 2. Computer programming class set-up with pair programming

2) Coding over Passive Listening: To achieve the second norm of the agile manifesto which is "*working software over comprehensive documentation*", it is valuable that students being in pairs should directly participate and practice whatever their teacher is demonstrating on the whiteboard rather than being passive listeners, copying what they don't understand as notes. (see Fig. 1) In this way, as students practice by coding directly, they will get to understand how the code operates on the IDE in terms of execution as well as memorize and retain the steps used in achieving the results. They will be able to master the coding, the language syntax, program running and debugging where necessary. For example, if the teacher says all variables in a program must be declared before being actually used, as in the code segment shown in Fig. 3.



```
..........
..............................
public static void main(String[] args) {
  a = 1;
  b = 1;
  System.out.println(a);
  w hile (b < 50) {
    System.out.print(b);
    b = a + b; // new b
    a = b - a; // new a
  }
................................
```

Fig. 3. Sample program

The only way for the students to understand the syntax and the statement as shown in Fig. 3 is by running the sample program code with an undeclared variables to see if it compiles or not and be able to debug it where necessary. Consequently, the practice of this nature will be able to improve their understanding, master the coding and apply it whenever they are writing their own programs.

3) Active Teacher/Tutor Involvement: The third agile norm which is "*customer collaboration over contract negotiation*" can be achieved with a close teacher/tutor contact throughout the CP class. Since it is not possible to have external customer in the students' learning environment to interact with them, the teacher having sufficient domain and programming knowledge will assume the responsibility of the customer. Though negotiation of contract is not applicable, the teacher being the instructor will teach, give the students programming tasks and maintain frequent contact with the students to see that they are doing the right thing as well as assisting them when they get stuck while performing their task. In addition, instead of submitting their solutions after performing their tasks, the teacher through active involvement would assess the students' work directly where applicable.

4) Content Flexibility and Immediate Corrections: The agile norm "*responding to change over following a plan*" can be achieved in two different ways in order to ensure that the students achieved their learning outcome. One way is by being able to adjust the course contents to the needs of the students when the need arise, especially if they are having problems of understanding that is not rooted from the teacher's experience. Basically, the first year students are new to programming and they are supposed to be taught based on their level of understanding instead of just following the course content directly as stipulated in the curriculum. In the event of students having problems of understanding in a particular topic, it should be adjusted to suit their level by strictly following the Bloom's taxonomy [28]. The other aspect that is also important is the immediate correction of programming errors during CP class. Since it is the duty of the customer (teacher) to maintain close contact with the students, thus, it is important that whenever students are found not doing the right thing, they should be noted and changes should be effected immdiately without having to wait for the work to be submitted to the teacher for assessment before changes are made. In this case, when required changes are made at the right time, it will save the effort and time to do rework on the part of students and assessments on the part of the teacher.

VI. Discussions

Teaching and learning in higher education worldwide is witnessing a shift in the paradigm ICT usage has brought about. In short, ICT is promoting new approaches to teaching and learning. However, due to issues of students' diversity, the current use of ICT is not enough to deal with certain issues that arise. Therefore, the application of the technology has to be improved. To achieve this, we believe that applying some cutting-edge ICT practices and processes like agile methods can be valuable in the enhancement of current teaching approaches. The application of agile principles and practices into the teaching of CP course will help, perhaps, the first year CS students some of whom have not seen, used or touch computer to get actively engaged and participate in CP classes. They will be able to learn how to operate the computer, program, and improve their programming skills through active team collaboration for a project success. On the part of the teacher, this approach will help them manage large classes effectively by getting students engage and participate in their learning process.

In particular, the incorporation of pair programming as one of the core practices in agile process has been commendable and productive. The rationale for its application in this proposed teaching methodology stems from several researches that have been carried out in the literature. For instance, it has been reported that pair programming have been used as learning method in several programming courses [18]. Bipp et al [20] in their studies found that pair programmer students can produce more code which are readable than their non-pair programmer counterpart. In a similar study by [19], the study found some elements of proficiency in individual programming activities and the retention of positive attitudes towards programming courses in terms of motivation for students who practiced pair programing. Another essential aspect of pair programming is the peer evaluation of each other's programming activities among students. Based on this, Gatfield [21] stated that as students assesses each other's work there is satisfaction and learning effectiveness resulting from the students. In addition, through pair programming, novice programmers are encouraged to improve their understanding. There are several other studies in the literature that are not discussed in this paper. With the relevance of the agile practices in these studies, our intuition in this paper is that incorporating agile practices into introductory CP course teaching will help to create a learning environment that is problem-based which is effective at helping students to engage in their actual learning of computer programming and to excel in their software development profession thereafter.

A. Limitations

As agile practices are gaining momentum in software development, it is of the essence that these practices are integrated into our teaching approaches as well. However, despite the fact that agile process can help increase students programming competencies, help under-prepared to achieve their learning outcome without posing a burden, a successful implementation of this idea will really not be easy to achieve as anticipated in most universities education. This is because it will not be easy to have a teacher or mentor next to the students throughout all the classes. In addition, in most higher education where lack of infrastructures and resources are experienced, it will be difficult if not impossible to implement the approach. Lastly, due to the nature of the class set-up and the role of the teacher as the customer, it could be difficult to cover the expected course contents for a term. This will pose a serious challenge to approach. But the goal of the university is really to increase computer science enrolment and the numbers of quality graduates that fits the need of software development organizations, efforts must be in place to make this idea a success. This is just creating the right environment that get students engaged as possible.

VII. Conclusions

In this paper we have presented a proposal on the new approach of teaching computer programming for beginners using the agile principles and practices of pair programming. Though several studies have been reported on the application of agile methods in students' project, courses or learning activities, no known study have incorporated active use this technique in the teaching of computer programming courses. Due to the difficulty involve in computer programming coupled with the nature of first year students, we believe it is important for teachers of these courses to align their teaching approach to the agile manifesto and pair programming practices. This involves having programming lectures in the computer-equipped classrooms with students sitting in pairs instead of teaching in the open classroom. By so doing, it will bring about improved understanding, active engagement and participation on the part of the students to achieve their overall learning goals. In the same vein, the approach will help the teacher to manage large classes which has been a serious problem in institutions of higher learning effectively.

Notwithstanding some of the limitations we have highlighted, implementing this idea will help eliminate factors impacting students' learning motivation of introductory computer programming courses. Thus, our future research will be on the implementation of this teaching methodology in order to validate its effectiveness.

REFERENCES

[1] Northedge, A. Rethinking Teaching in the Context of Diversity Teaching in Higher Education, Vol. 8, No. 1, 2003.

[2] Biggs, J. What the Student Does: teaching for enhanced learning, Higher Education Research & Development, 18:1, 57-75, 1999.

[3] Jones, G. and Knezek, G. Non-commercial radio-satellite telecommunications: affrordable options for technology educators. Cited in S. Romi (2000) Distance Learning and Non-formal Education: Existing Trends and New Possibilities of Distance Learning Experiences. Educational Media International. 37 (1), 39-44, 1993.

[4] Perera, G.I.U.S. Impact of using agile practice for student software projects in computer science education. International Journal of Education and Development using Information and Communication Technology (IJEDICT) Vol. 5, Issue 3, pp. 85-100, 2009.

[5] Gunter, G. A. "The Effects of the Impact of Instructional Immediacy on Cognition and Learning in Online Classes", International Journal of Social Science , vol. 2 no. 3, pp. 196-202, 2007.

[6] Dede, C. Learning about teaching and vice versa. Paper presented at Conference of Society for Information Technology in Education. Washington D.C., USA, 1998.

[7] Forcheri, P. and Molfino, M. T. ICT as a tool for learning to learn. In Watson, D. M. and Downes, T. (Eds.) Communications and Networking in Education. Boston, MA: Kluwer Academic. pp 175-184, 2000.

[8] Kris M.Y., Victor C.S., Lee Y.T. Yu. Learning motivation in e-learning facilitated computer programming courses. Computers & Education 55, pp. 218–228, 2010.

[9] Deek, F.P., & McHugh, J. Problem Solving and Cognitive Foundations for Program Development: An Integrated Model. Proceedings of the Sixth International Conference on Computer Based Learning in Science (CBLIS), Nicosia, Cyprus, pp. 266- 271, 2003.

[10] Lam, M. S. W., Chan, E. Y. K., Lee, V. C. S., and Yu, Y. T. (2008): Designing an automatic debugging assistant for improving the learning of computer programming. Lecture Notes in Computer Science, 5169, 359–370.

[11] Foster, A. 2005. Student interest in computer science plummets. Chronicle of Higher Education. http://chronicle.com/free/v51/i38/38a03101.htm. (Accessed on August 31, 2006).

[12] Wulf, T. Constructivist approaches for teaching computer programming. SIGITE'05, Newark, New Jersey, USA. October 20–22, 2005.

[13] Wells, G. & Claxton, G. Learning For Life in the 21st Century: sociocultural perspectives on the future of education (Oxford, Blackwell), 2002.

[14] Perera, G.I.U.S. and Fernando, M.S.D. 2007, "Enhanced Agile Software Development — Hybrid Paradigm with LEAN Practice, In Proc. of 2nd International Conference on Industrial and Information Systems, ICIIS 2007, IEEE , pp. 239 – 244.

[15] Highsmith, J., Cockburn, A., 2001, "Agile Software Development: The Business of Innovation", IEEE Computer, vol. 34, pp.120-127.

[16] F. Maurer and S. Martel, "Extreme programming: Rapid development for web-based applications," IEEE Internet Comput., vol. 6, no. 1, pp. 86–90, Jan./Feb. 2002.

[17] Manifesto for Agile Software Development: http://www.agilemanifesto.org.

[18] Williams, L., McDowell, C., Nagappan, N., Fernald, J., and Werner, L. "Building pair programming knowledge through a family of experiments", In Proc. of International Symposium on Empirical Software Engineering, ISESE 2003, pp. 143-152, 2003.

[19] Williams, L., Yang, K., Wiebe, E., Ferzli, M., and Miller, C. "In Support of Pair Programming in Introductory Computer Science Course", Computer Science Education, vol. 12(3), pp.197–202, 2002.

[20] Bipp, T., Lepper, A., and Schmedding, D. "Pair programming in software development teams – An empirical study of its benefits, Information and Software Technology, 50, pp. 231–240, 2008.

[21] Gatfield, T. "Examining Student Satisfaction with Group Projects and Peer Assessment". Assessment and Evaluation in Higher Education, Vol. 24(4), pp. 365-377, 1999.

[22] Weber, G.; Brusilovsky, M. S.; Steinle, F. ELM-PE: An Intelligent Learning Environment for Programming, 1996. Disponivelem<http://www.psychologie.uni-trier.de: 8000/projects/ELM/elm.html>. Acesso em: 2014.

[23] Gomes, A., & Mendes, A. J. Learning to program – Difficulties and solutions. In International Conference on Engineering Education – ICEE 2007, Coimbra, Portugal, 2007.

[24] Jenkins, T. On the difficulty of learning to program, In: 3rd annual conference of LTSN-ICS. 2002.

[25] Lui, A. K., Kwan, R., Poon, M., and Cheung, Y. H. Y (2004). Saving weak programming students: Applying constructivism in a first programming course, SIGCSE Bulletin, 36, pp.72–76, 2004.

[26] Xue Xue, "Research and Practice of Teaching at Different Levels", Xiangtan Normal University Journal, Xiangtan, Hunan, 2008(4), pp. 226-227.

[27] Rolandsson, L. Changing computer programming education: the dinosaur that survived in school. An explorative study about educational issues based on teachers' beliefs and curriculum development in secondary school. The 2013 Learning and Teaching in Computing and Engineering, IEEE. 2013.

[28] Krathwohl, D. R. A revision of Bloom's taxonomy: an overview. Theory into Practice 41, p. 212-218. 2002.

Isong E. Bassey received B.Sc. degree in Computer Science from the University of Calabar, Nigeria in 2004 and M.Sc. degree in both Computer Science and Software Engineering from Blekinge Institute of Technology, Sweden in 2008 and 2010 respectively. Presently, he is a Ph.D candidate in Computer Science Department of North-West University, South Africa since 2012. Since 2010, he has being a faculty member of the University of Venda, South Africa and a lecturer of Computer Science and Information Systems. His research interests include Requirements Engineering, Software Maintenance, Information Security, Software Testing, and Mobile Computing.