Integration Of Innovative Technologies And Affective Teaching & Learning In Programming Courses

Alvin Prasad, Mohammed Farik

Abstract: Technology has been integral component in the teaching and learning process in this millennium. In this review paper, we evaluate the different technologies which are used to currently facilitate the teaching and learning of computer programming courses. The aim is to identify problems or gaps in technology usage in the learning environment, and suggest affective solutions for technology integration into programming courses at the University levels in the future. We believe that with the inclusion of suggested innovative technologies and affective solutions in programming courses, teaching and learning will be attractive and best for the programming industry.

Index Terms: Affective, Education Technology, Programming Course, Teaching & Learning

————————————◆————————————

1 Introduction

PROGRAMMING is an important subject in *Computer Science* and *Information Systems* developments, and its demand in the computer industry is even greater today. New applications are being coded and existing applications recoded. The *mobile* and *smart phone* market is thriving today and users need related applications for their everyday routines. Moreover, the software development industry needs capable programmers who have the in-demand programming language skills. Likewise, businesses such as *Cloud Service Providers* need programmers for *remote-procedure call* (RPC) programming in their distributed network servers, to fine-tune operating systems for *PaaS* and develop software for *SaaS* clients. However, there is shortage of programmers because of various reasons, one of which is that learning programming is not an easy task. Thus, Colleges and Universities need to integrate innovative technology and affection in their programming courses like they have never used before to generate interest and motivate students into becoming quality programmers. In this review paper we will evaluate some latest technologies and relevant gaps in teaching and learning computer programming in section 2, and introduce the affective domain of learning in section 3. Next we suggest some solutions as our contribution in section 3 and finally conclude in section 4.

2 Innovative Technologies

Previously, knowledge was communicated in a classroom environment with the use of black board, hard cover books, charts, chalk and a teacher. In current time technology is incorporated in the traditional setting where interactive boards, projectors, internet access, mobile devices and teacher's technological toolkit are used which motivates and enhance teaching and learning [1]. Teacher's toolkit includes serious games, animations, quality images, videos, blog, web addresses and educational applications. Future of education predicts virtual learning [2], mobile learning [3], games [4] with digitalized classrooms and digital books. This digitalization allows students to collaborate with peers globally, thus providing them with more information and enhanced understanding. With the rise in use of mobile and smart phones, there is a greater demand for mobile applications development and related programming skills. There is high demand for suitable software applications in the fields of education, business, medical, government, transportation, tourism and other sectors, and hence the need for more programmers with knowledge of latest programming platforms and languages. The onus is on the universities to prepare quality programmers with marketable programming skills for today's job market. Krpan *et al.* [5] and Woei *et al.* [6] argue that teaching and learning computer programming is challenging. Students face a lot of difficulties in trying to understand computer programming while teachers are trying to help students understand different programming languages. Researchers suggest different technologies for teaching and learning of computer programming at different stages like novice, intermediate and in advance software engineering. Different technologies can be integrated to help in effective teaching and learning in programming courses at College and University levels.

- *Alvin Prasad is currently pursuing masters degree program in information technology in the School of Science and Technology at The University of Fiji. Email: alvinp@unifiji.ac.fj*
- *Mohammed Farik is a Lecturer in Information Technology in the School of Science and Technology at The University of Fiji.*
- *Email: mohammedf@unifiji.ac.fj*

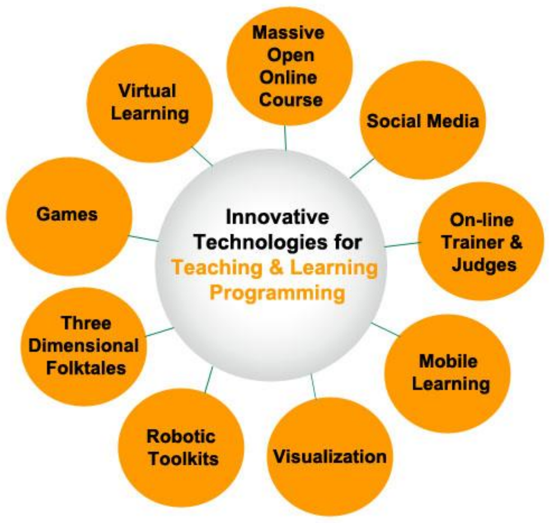

Massive Open Online Course
Virtual Learning
Social Media
Games
Innovative Technologies for Teaching & Learning Programming
On-line Trainer & Judges
Three Dimensional Folktales
Mobile Learning
Robotic Toolkits
Visualization

Fig.1 *Innovative Technologies for Teaching and Learning Programming*

2.1 Virtual Learning

One of the latest technologies getting recognition from the researchers in the education sector is Virtual world. One of the virtual world platforms that are used by learners and educators is *Second Life* (SL). Esteves *et al.* [7] argue that SL could be used to teach programming at university level but they did not give any clear guidelines as to how it can be implemented. They also identified three issues and suggested solutions. First issue is in relation to communication between teacher and students. It was not clear how the teacher could explain subjects and clarify student's doubts within SL. So, the authors experimented with voice-based and text-based SL communication. Voice based communication was not used as it required an environment with less amount of noise. Text based communication was tried which has public and private channels where it retains a history which can be revisited as and when needed. Public channel was used first but under public channel of text based learning the communication can be seen by everyone who is within 20m of virtual world space. They found that it is difficult to use public channel as all the communication appears on the screen from different sources and it becomes very difficult to manage. Then they followed the private communication channel where communication is between two people, this solved the issue of congestion on the screen and students were happy as they were talking to a teacher directly were no one is reading their messages but another issue which was identified in the research is that now the teacher finds it difficult to respond to each of the questions quickly so the feedback time is delayed. To solve the feedback issue it was suggested that teachers can keep pre-prepared short messages which can be copied and pasted to save time. The second issue is in relation to student's process of learning where they tried to find how the process of learning takes place. In the research, project was used as the starting point for the students learning, but the problem was on the type of project. To solve this issue, visual projects were used which was preferred by the students. The last issue is the teaching process. It was found that teachers work is more difficult compared to traditional way of teaching. Here the teachers need to be very active and be well prepared. To solve this issue the authors suggested that there should be some procedure where the students can inform the teacher well in advance on the problems which they faced.

2.2 Massive Open Online Course

Furthermore, Massive open online course (MOOC) is also developed by universities to teach programming. It is one of the latest online teaching and learning model Liyanagunawardena *et al.* [8] which is gaining a lot of popularity. As discussed by Eckerdal *et al.* [9], people have mixed feelings about MOOCs and it has positive and negative aspects. Some of the good features are that it is affordable, accessible anywhere and anytime and allows interaction with experts. The negative aspects are that it does not have a clear teaching structure and does not have a satisfactory grading system which leads to learners not getting the correct feedback. To solve the issue of instant feedback researchers Tillmann *et al.* [10] propose a gaming platform *Pex for fun* (Pex4Fun). It is a browser based game which can be used on any browser from any device. It provides instant feedback to students and comes with auto-completing code editor. The key idea behind this platform is a sample solution which acts as a game where the student learner tries to complete the given code to match the sample solution by adding the codes and learning at the same time.

2.3 3-Dimensional Folktales

Another technique to learn programming suggested by Woei *et al.* [6] is using objects-first approach through folktales. The authors propose that to help in learning programming, as they also find that programming is really challenging, folktales should be used. They claim that folktales will motivate novice programmers and non programmers towards programming as they will have a clear understanding of the story. In this research students were asked to develop folktales using 3-dimensional (3D) application software *Alice* and *Unity*.

2.4 Online Trainer and Judges

Verdú *et al.* [11] state that many online programming trainers have been developed and are available online to help in teaching and learning programming but there is no proper teaching functionalities or standards. *UVA online judge* is an online programming trainer which has several limitations. According to Rongas *et al.* [12], introductory programming learning tool can be classified into four categories: Integrated development interface, visualization, virtual learning environment & submission, managing and testing system. Verdú *et al.* [11] suggest that online teaching tools should have at least a combination of virtual learning environment and a submission system. To address the issue of online trainers and judges, and to incorporate the features suggested, a project *EduJudge* is developed. EduJudge is a distributed system for learning programing which is composed of three main subsystems: an evaluation server (*UVA On-line judge*), a learning objects repository (*crimsonHex*) and a user interface (consists of a set of *plugins* and *modules* for *Moodle*). Studies suggest that automated assessment and competition increase interest in students and motivates them to learn. So EduJudge have both the features incorporated.

2.5 Visualization

Linden & Lederman [13] argue that many tools are available but its usage is very limited, time consuming, complex and expensive. Therefore, they suggest that instructors use visual tools to teach programming. They propose reuse of the existing tools according to the concepts that needs to be taught, customize the tools according to the student's interest and minimize the binding of the concepts that the instructors are teaching. There should be minimal dependencies between each of the learning concepts.

2.6 Robotic Toolkits

Sullivan *et al.* [14] proposes the use of physical and tangible tools like robot to engage young children in programming learning. In their research, they developed a robotic kit to be used by children aged 4 to 7 to learn the concept of programming. They found that there are lot of benefits of using tangible objects in early education that enhances the skills of students towards mathematics, programming and engineering.

2.7 Games

Many different games such as *Colobot* and *Robocode* [15] have been developed to teach computer programming concepts. It was found that games motivate students towards programming, helps in building the skills of problem-solving, and helps students to think critically. But it also has some limitations like some students do not enjoy the game [16] and leave it without completing, which can lead to students discouraged about learning programming. This was resolved in the research through motivating students to use the tutorials. Another challenge which is faced by some serious game developers is to create games which can be played on mobile devices. To solve this issue, many researches are in progress [17]. Together with this, researchers Leutenegger & Edgington [18] propose that games programming should be taught at the early stages of learning programming as this concept will motivate students, which will assist them to visualize what they have developed.

2.8 Mobile Learning

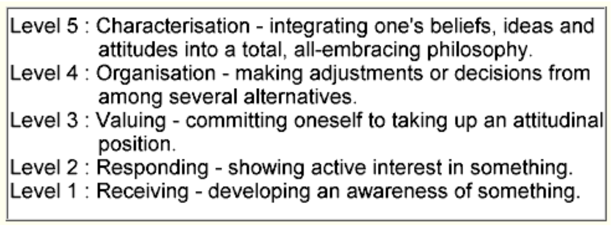
In 2012, Tillmann *et al.* [19] suggest that in the future, programming will be taught via mobile phone. Moreover, graphics programming are being taught using android mobile devices [20]. There were some drawbacks of using this approach like the amount of time spent on preparing the device suitable for teaching, heterogeneous platforms, user interface and smaller screen. Some issues were solved with high resolution and *Pan* and *Zoom* features.

2.9 Social Media

Researchers have found that social media can also be incorporated to facilitate computer programming education. According to Hew [21], social media can be utilized by students and teachers to communicate problems, ideas and findings. It is also evident that while using social media students and teachers will be able to collaborate with the international world. Bitar & Melki [22] explain the use of *Elgg* as an open source social networking site in teaching programming. Their research shows that students and teachers both are satisfied with the features of Elgg. Dzvapatsva *et al.* [23] show that it increases the pass rate in computer programming courses. However, there are some disadvantages like bullying, rude conversations, wrong solutions to questions. Their advice is to control all this with standards and policies.

3 Affective Domain

The affective domain highlights teacher and learner attitudes, feelings, and empathy in delivery of learning objectives. Fig. 4 describes the affective domain levels beginning with *receiving*, *responding*, *valuing*, *organization*, and finally *characterization*. This shows that if teachers are able to create awareness from the onset, students will be able to carry on through the five levels. This way student will not lose interest towards programming. Law *et al.* [24] suggest that *intrinsic factors* such as attitudes, expectations, goals and emotions as well as *extrinsic factors* like clear direction, reward, recognition, punishment, social pressure and competition can motivate students towards programming. Also, visualization using the many tools mentioned in this section (games, mobile learning, MOOCs, virtual world), creates a positive impact on students learning. However, Musa *et al.* [25] states that to create a greater impact via these tools, the different personality type (psychology, colour science, type of animation, different shapes) of humans should be considered. Through this survey they found out that considering the personality types of students' temperament in designing educational computer animations helps to enhance the learning process.



Level 5 : Characterisation - integrating one's beliefs, ideas and attitudes into a total, all-embracing philosophy.
Level 4 : Organisation - making adjustments or decisions from among several alternatives.
Level 3 : Valuing - committing oneself to taking up an attitudinal position.
Level 2 : Responding - showing active interest in something.
Level 1 : Receiving - developing an awareness of something.

Fig. 2 *The Bloom/Krathwohl hierarchy of the affective domain (1964)*
http://www2.rgu.ac.uk/celt/pgcerttlt/specifying/speci6.htm

4 Our Contribution

The over-use of technology has somewhat also aided in removing the personal touch from our lives. We suggest implementing this personal touch, the affection, motivations in the use of technology and in teaching & learning difficult subjects like programming. We believe students should be affectively motivated to appreciate the importance of programming in our daily lives. They should be able to realize that programming solves problems. This is where visualization can come in handy. Programming language like *Scratch* can be used by students to build simple logic games and solve simple problems to generate interest in programming before exposing students to more serious languages such as *C++* and marketable skills such business application development. Furthermore, whether the learning takes place in virtual environment or online, an instructor's presence is very important. This is so that there is no communication-breakdown in teacher feedbacks to student queries. Clear instructions, timely and detailed feedback by the instructors are very crucial at every stage of program learning. The teacher needs to maintain the flow of thought processes in real-time so that the student does no give-up the programming challenge. Furthermore, hands-on or interactivity is very important. During problem understanding, instructors should explain a similar question by doing the coding. Then allow the students to work on that on their own. This learning can be online or offline which should be user-friendly. Moreover, more of *Serious games* type of technology that allows interactivity where learners feel the ownership of their developed code, should be used in the courses. We suggest application developers should develop more of these games for mobile devices, despite the challenges of heterogeneous platform. These games should have user-friendly interface, and programmed for different levels of learners, should be playable in competition with a friend, and be fairly graded with scores. Also, a cost effective environment, either inside the computer laboratories or places where students hang out in the campus, should be created for students to be actively involved with programming tools via learning technologies. Additionally, while group-learning and peer learning should be very much encouraged, learners should also be allowed to learn individually as well. Technology choice for implementation in this regard should consider various interest groups, which is different types of learners with different capabilities. Learners should be allowed to learn at their pace. They should get visual tutorial and notes for individual concepts. The visual tool should include real field examples with sample demonstration of codes and output solutions or onscreen effects. Likewise, utilization of social media equipped with effective policies, protocols, security, and privacy can be instrumental for effective formal and informal discussions with peers and the international community. Finally, we would like to suggest engaging learners with the relevant organizations for real-project industry programming attachments. This involvement will allow leaners to gauge their skills against industry benchmarks, and self-motivate themselves to perform accordingly. Such industry experience will also be advantages to them in searching for jobs after graduating.

5 Conclusion

In conclusion, in this review we have given an overview of the innovative tools that is used in teaching and learning computer programming. We have stated the problems in educational implementation of each technology in the literature, and gave suggestions affective implementations as solutions for producing quality graduates. These suggestions can be incorporated in thought processes while making technology choice for implementation, while designing programming course structure and lesson plans, and during lesson delivery in the teaching & learning environment.

References

[1] J. Wakefield. (2015, February) BBC News. [Online]. http://www.bbc.com/news/technology-30814302

[2] L.G.R Rolando, D.F. Salvador, A.H.S. Souza, and M.R.M.P. Luz, "Learning with their peers: Using a virtual learning community to improve an in-service Biology teacher education program in Brazil," Elsevier, 2014.

[3] M. Ally and J. Prieto-Blázquez, "What is the future of mobile learning in education?," Universities and Knowledge Society Journal, pp. 142-151, 2014.

[4] J. Hamari, K. Jonna, and H. Sarsa, "Does Gamification Work? — A Literature Review of Empirical Studies on Gamification," in 47th Hawaii International Conference on System Science, 2014, pp. 3025-3034.

[5] D. Krpan, S. Mladenović, and M. Rosić, "Undergraduate Programming Courses, Students' Perception and Success," in International Conference on New Horizons in Education, 2015, pp. 3868–3872.

[6] L.S. Woei, I.H. Othman, and C.K. Man, "Learning programming using objects-first approach through folktales," Penerbit UTM Press, pp. 47-53, 2014.

[7] M. Esteves, B. Fonseca, L. Morgado, and P. Martins, "Improving teaching and learning of computer programming through the use of the Second Life virtual world," British Journal of Educational Technology, 2011.

[8] T.R. Liyanagunawardena, A.A. Adams, and S.A. Williams, "MOOCs: A systematic study of the published literature 2008-2012," The international review of research in open and distributed learning, pp. 202-227, 2013.

[9] Eckerdal, P. Kinnunen, N. Thota, A. Nylen, J. Sheard, L. Malmi, "Teaching and learning with MOOCs: computing academics' perspectives and engagement," in Proceedings of the 2014 conference on Innovation & technology in computer science education, New York, 2014, pp. 9-14.

[10] N. Tillmann, J.D. Halleux, T. Xie, S. Gulwani, and J. Bishop, "Teaching and learning programming and software engineering via interactive gaming," in ICSE, San Francisco, CA, 2013, pp. 1117 - 1126.

[11] E. Verdú, L.M. Regueras, M.J. Verdú, J.P. Leal, J.P.D. Castro, R. Queiros, "A distributed system for learning programming on-line," Elsevier Ltd, pp. 1-10, 2011.

[12] T. Rongas, A. Kaarna, and H. Kalviainen, "Classification of computerized learning tools for introductory programming courses: learning approach," in In Proceedings of the IEEE international conference on advanced learning technologies, 2004, pp. 678 - 680.

[13] T. Linden and R. Lederman, "Creating Visualizations from Multimedia Building Blocks: A Simple Approach to Teaching Programming Concepts," in Information Systems Educators Conference, 2011, pp. 1-10.

[14] Sullivan, M. Elkin, and M.U. Bers, "KIBO Robot Demo: Engaging Young Children in Programming and Engineering," in Proceedings of the 14th International Conference on Interaction Design and Children, New York,

2015, pp. 418-421.

[15] N. Tillmann, J.D. Halleux, and T. Xie, "Pex4Fun: Teaching and Learning Computer Science via Social Gaming," in 2012 IEEE 25th Conference on Software Engineering Education and Training (CSEE&T), Nanjing, 2012, pp. 90-91.

[16] C. Liu and Y. Huang, and C. Cheng, "The effect of simulation games on the learning of computational problem solving," Computers & Education, pp. 1907–1918, 2011.

[17] T. Jordine, Y. Liang, and E. Ihler, "A mobile-device based serious gaming approach for teaching and learning Java programming," in IEEE Frontiers in Education Conference (FIE) , Madrid, 2014, pp. 1-5.

[18] S. Leutenegger and J. Edgington, "A Games First Approach to Teaching Introductory Programming," in Proceedings of the 38th SIGCSE technical symposium on Computer science education, New York, 2007, pp. 115-118.

[19] N. Tillmann, M. Moskal, J.D. Halleux, M. Fahndrich, J. Bishop, A. Samuel, T Xie, "The Future of Teaching Programming is on Mobile Devices," in Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education, New York, 2012, pp. 156-161.

[20] M. Werner, "Teaching graphics programming on mobile devices," Journal of Computing Sciences in Colleges, pp. 125-131, 2013.

[21] K.F. Hew, "Students' and teachers' use of Facebook," Computers in Human Behavior, pp. 662-676, 2011.

[22] Bitar and A. Melki, "Open Source Social Technologies for Teaching Computer Programming," International Journal of Recent Development in Engineering and Technology, pp. 59-63, 2014.

[23] G.P. Dzvapatsva, G. Whyte, and Z. Mitrovic, "Social media as a tool for improving the pass rate in computer programming for FET students," in ReSNES 2011 Research Colloquium, 2011.

[24] K.M.Y Law, V.C.S Lee, and Y.T. Yu, "Learning motivation in e-learning facilitated computer programming courses," Computers & Education, pp. 218–228, 2010.

[25] S. Musa, R. Ziatdinov, and S.F. Omer, "Developing Educational Computer Animation Based on Human Personality Types," European Journal of Contemporary Education, pp. 52-71, 2015.