Volume 1 No. 8, December 2011 ISSN-2223-4985
International Journal of Information and Communication Technology Research

http://www.esjournals.org

Using Program Visualization to Improve ICT skills towards achieving Vision 2030

[1]Mutua M. Stephen, [2]Wabwoba Franklin, [3]Ogao J. Patrick, [4]Abenga Elizabeth, [5]Juma Kilwake
[1, 2, 4, 5]Masinde Muliro University of Science & Technology,[1, 2, 5]Department of Computer Science,[4]Department of Curriculum and Instruction, [3]Kenya Polytechnic University College

ABSTRACT

ICT has become an imperative component in enhancing industrialization and socio-economic growth in developed and developing countries. It is thus irresistible for any nation that wishes and plans to expand its growth. In the Vision 2030, the country seeks to provide ICT services that are quality and affordable to many. Any ICT component will include some software (program) and hardware. However, programming remains a hurdle to many ICT professionals, teachers and the students thereof and thus country imports most of its software. This poses a danger of slowing down possible innovations and developments that would otherwise be realized if the programming skill is mastered. This paper evaluates programming from the students' perspective on its relevance, understandability and usage of emerging trends to learn it. During the study, a sampled population of students of IT and computer science students was evaluated using questionnaires. It was evident that most students desired better ways other than the chalk-board method or using PowerPoint slides. It however emerged that most students were unaware of existing tools that can aid learn programming. The paper culminates by a brief discussion of the emerging trend of using program visualization tools to teach/learn programming.

Keywords: *Program Visualization, Programming, Software visualization*

1. INTRODUCTION

ICT is forming part of our daily lives. All the spheres of life are in one way or another being fuelled by the use ICT. The government of Kenya promulgated the ICT policy in 2006 with the goal of improving the livelihoods of the citizens through the availability of accessible, efficient, reliable and affordable ICT services [1]. To offer these services efficiently requires a team of qualified and knowledgeable ICT professionals. The achievement of this is only possible if the to-be professionals learn and practice the skill well, and then effectively apply it in the field. Programming is the art of writing codes using a certain language to instruct the computer on what to do. This is such an important skill that if well nurtured can propel the nation highly towards achieving Vision 2030. This will be through the provision of locally made, available, affordable powerful software that will cut down on the cost spent in importing Computer Software. Due to its importance, this is why one of the key courses in any ICT programme is *programming*. It is so crucial that a student who fails to take it during his/her course work is perceived to be 'half-baked' in the professional world. However, despite its importance, this course has proved to be an uphill task to many students. Similarly, instructors also find it challenging to impart this important skill [2] to students hence the need to change the tactic of how the course is taught. Various world renowned universities have moved steps further in improving the teaching using program visualization (PV) tools.

This paper is as a result of some work which is ongoing in which the researchers are evaluating various PV tools in a goal of providing a proper guideline to both teachers and student on these tools world over. We present some preliminary results which we conducted in establishing the usage and awareness of these tools to students in MMUST.

The rest of the paper is structured as follows: Section 2 reviews the literature and prior related research work. Section 3 focuses on data collection procedure, results from the student responses as well as the discussion. Section 4 discusses program visualization as an emerging trend of teaching programming. Finally, Section 5 concludes the paper with a summary and overview of some tools that we are in the process of evaluating.

2. LITERATURE REVIEW

To understand the potential, development and the limitations of the smart computer systems of the future, one must first come to terms with the basic vocabulary of such systems which is provided for by a programming language [3]. [4] assert that "understanding the design, code, and behavior of software systems is thus an essential and important problem'' and that's why as human's we use pictures to enforce understanding in various facets of life. This may be because as the old adage puts it, *a picture is worth a thousand words.* This is the motivation of using PV to enhance understanding in programming classes.

Kristi [5] analyses various difficulties highlighted by various researchers of learning and teaching programming. Some of the common problem posed by many novice students is that they find difficult in comprehending the abstract features of programming and

relating them to real life objects. This seems to be supported by observations made by various researchers [6][9].

The trials of these tools in assorted Universities and other institutions have posted positive results as they have shown significant improvement of performance by various students [7] [8]. However, despite their positive postings, they have not being widely used as anticipated. This seems to worry researchers why they work so hard yet the tools are not used [10]. In Kenya, we tend to agree that it due to lack of awareness of the availability of these tools as our research shows.

This work will discuss these further and will act as an awareness and call to students and teachers to heed to the changing times to enhance understandability in class; as well as producing professionals who will propel the nation towards achieving its Vision 2030 technologically.

3. DATA COLLECTION, RESULTS AND DISCUSSION

3.1 Data Collection

During our study, we issued questionnaires to students of Computer Science (CS) and Information Technology (IT) who we considered to be leading future ICT professionals; all within the Computer Science Department (CSD). The questionnaire comprised of open-ended questions and closed-ended questions. We used the

Likert Scale in the closed-ended questions and thus analyzed the data using frequency counts and graphs. We then used thematic analysis technique to analyze the open-ended questions.

Using the various class enrollments, we calculated $^1/_3$ of each of the study years (1 – 4) and issued the questionnaires. This resulted into the following sampled population:

Table I – CS Undergraduate sample population: Source CSD

Target Population			**Accessible Population**		
	Population Size		**Fraction**	**Sample Size**	
Study Year	*Male*	*Female*		*Male*	*Female*
First	56	6	$^1/_3$	19	2
Second	55	10	$^1/_3$	18	3
Third	44	3	$^1/_3$	15	1
Fourth	42	8	$^1/_3$	14	3
Sub-Total	**197**	**27**	$^1/_3$	**66**	**9**
Total		**224**	**Total**		**75**

We also issued the questionnaires to IT students whose samples are shown in table II below;

Table II – IT Undergraduate sample population: Source CSD

Target Population			**Accessible Population**		
	Population Size		**Proportion**	**Sample Size**	
Study Year	*Male*	*Female*		*Male*	*Female*
First	47	12	$^1/_3$	16	4
Second	15	10	$^1/_3$	5	3
Sub-Total	**62**	**22**	$^1/_3$	**21**	**7**
Total		**84**	**Total**		**28**

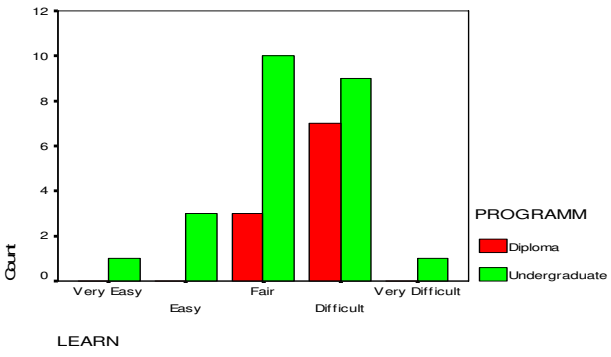
Finally, we also collected data from the Diploma in IT (DIT) second year students. The whole population was small and accessible hence all the ten (10) students presented their data.

By the time we were writing this paper, we had collected and analyzed data from all the second year students in both CS and IT, as well as the DIT students. Since, this is an ongoing research work as earlier stated; we write this paper based on the results so far analyzed whose results have indicated some worthwhile trend.

3.2 Results and Discussion

From the 29 questionnaires issued to second year students (21 from CS and 8 from IT), 24 questionnaires were received back. This represented 82.76% response which we considered trustworthy. In our instruments we desired to know how the respondents rated learning programming, how they perceived its relevance and whether they felt that there was need for use of other tools other than chalk and board. Ultimately we also desired to analyze the awareness and usage of PV tools among other demographic information.

Learning Programming

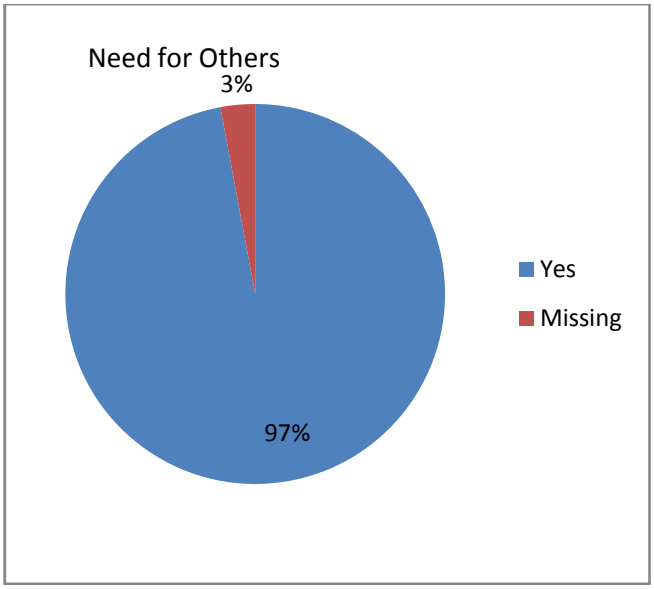


From this the respondents (students) confirmed what many researchers have said. They thus rated learning programming as an uphill task. However, most of the undergraduate students rated it fairly. This thus seems to be a problem that has been persistent over years as proved by Soloway and Sphorer [11] in their paper written some decades ago.

Relevance

Despite the difficult portrayed above, they unanimously agreed that programming was relevant to their programme and career path. 52.9% rated it as very relevant, 44.1% viewed it as relevant while a dismal 2.9% said that it was fairly relevant.

Need for other tools



Out of all respondents an overwhelming 97% concurred that there was dire need for other tools to aid in learning and understanding programming.

Usage of PV tools

Several respondents seemed unaware of what PVs are. For instance, despite that the DIT students indicated having used a PV, this was proved wrong by the examples they cited. Most indicated that they had used Visual Basic which is a Visual programming language and not a PV. Most second year students agreed to have used BlueJ a PV. They however seemed to be unaware of any other existing PVs. This proved that again they were unaware of what PVs are and thus quoted BlueJ a tool that had been used by one of the researchers to introduce object-oriented programming to them; and had been enlisted in the questionnaire. Others cited JCreator (a Java IDE) as a PV which is not the case. This proved that PVs were unknown to most students in developing countries.

4. PROGRAM VISUALIZATION THE EMERGING TREND

PV is a Software Visualization (SV), which is defined as the use of typography, graphic design, computer animation and graphics technologies to facilitate the understanding of software" [13]. PV tools focus on explaining the execution of computer programs (a set of instructions) hence facilitating an access to dynamic and usually hidden processes during program run-time [12]. They enable students to interact with the tool using visual cues that be easily comprehended and enable one to understand the code.

The consistent use of PVs tools has proved to be very beneficial and of great use to weak students who can use the tool(s) outside the classroom. For instance, [14] found that students who actively used the Jeliot program animation system improved their learning results compared with a control group that did not use Jeliot. Research by [15] showed that program visualization increases the attention of students to the material being taught. These among other research results have posted positive results which can be transferred to the Kenyan Universities.

5. CONCLUSION & RECOMMENDATION

From this work, we conclude that PVs are very useful in teaching programming to novice and advanced students. Various abstract issues can be easily explained using PVs. We also have concluded that the usage of PVs is still minimal and most students are unaware of their existence.

We therefore recommend the embracing of these vital tools towards ensuring quality education and skill for the ICT future professionals. If well used, this will result into time saving, improved performance, confidence and quality professionals. This will ultimately provide for quality software which is affordable thus edging closer to Vision 2030 technologically.

The future work seeks to explore more on PVs in developing countries and provide a well laid guideline to both teachers and students. This will be through evaluation of several tools which include and not limited to JEliot3, BlueJ, Ville, ALICE, JEROO, Scratch among others all which are freely available.

REFERENCES

[1] "National ICT Policy." 2006. Ministry of Information and Communications

[2] *Using JEROO to Introduce Object-Oriented Programming*

[3] Avinash, K. *Teaching Programming,* Purdue University - http://cobweb.ecn.purdue.edu/~kak/programming.pdf

[4] Pauw, D, W., Reiss, P, S. and J, T. Stasko (2001), *ICSE Workshop on Software Visualization,* IEEE

[5] Kirsti, A., *PROBLEMS I N LEARNING AND TEACHING PROGRAMMING - a literature study for developing visualizations in the Codewitz-Minerva project,* Institute of Software Systems, Tampere University of Technology, Finland

[6] Robins A, Rountree, J. & Rountree, N. (2003), Learning and teaching programming: A review and Discussion, *Computer Science Education, Vol. 13-No. 2, pp. 137–172, 2003*

[7] Kasurinen, J., Mika, P. & Uolevi, N. (2008), A Study of Visualization in Introductory Programming, *PPIG, Lancaster 2008*

[8] Kouznetsova S, (2007), Using Bluej and Blackjack to Teach Object-oriented Design Concepts In Cs, *Department of Computing Science Sam Houston State University, Journal of Consortium for Computing Sciences in Colleges April 2007 (pp. 49 -55)*

[9] Sanders, D. & Dorn, B. (2003), Jeroo: A Tool for Introducing Object-Oriented Programming, *ACM SIGCSE'03, February 19-23, 2003*

[10] Bassat, L. & Mordechai, B. (2007), We Work So Hard and They Don't Use It: Acceptance of Software Tools by Teachers ITiCSE'07, *June 23–27, 2007, Dundee, Scotland, United Kingdom Copyright 2007*

[11] Soloway, E. & Spohrer, J. (1989), *Studying the Novice Programmer*, Lawrence Erlbaum Associates, Hillsdale, New Jersey. Pp. 497

[12] Bednarick, R., Moreno, A., Myller, N., & E, Sutinen (2005), Smart Program Visualization Technologies: Planning a Next Step, *Proceedings of the Fifth IEEE International Conference on Advanced Learning Technologies (ICALT'05)*

[13] Guest Editors' Foreword (2002), Software Visualization, *Journal of Visual Languages and Computing (2002) 13, 257 – 258*

[14] Bassat Levy, R., Ben-Ari, M., & Uronen, P. A. (2003), The Jeliot 2000 program animation system, *Computing Ed. 40, 1, 15–21*

[15] Ebel, G. & Ben-Ari, M. (2006), Affective effects of program visualization, *Proceedings of the 2nd International Computing Education Research Workshop (ICER'06). ACM Press, 1-5*