International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org, editorijaiem@gmail.com
Volume 1, Issue 4, December 2012 ISSN 2319 - 4847

Choosing Tools of Pedagogy (Case of Program Visualization)

Mutua Stephen[1], Abenga Elizabeth[2], Patrick Ogao[3], Wabwoba Franklin[4] and Anselmo Ikoha[5]

[1]Department of Computer Science. Masinde Muliro University of Science and Technology, Kenya

[2]Department of Curriculum and Instructional Technology, Masinde Muliro University of Science and Technology, Kenya

[3] Department of Computer and Information Technology, Kenya Polytechnic University College, Kenya

[4] Department of Information Technology, Kibabii University College, Kenya

[5] Department of Computer Science. Masinde Muliro University of Science and Technology, Kenya

ABSTRACT

Computer aided learning has over time been integrated in traditional pedagogical approaches in order to improve the quality of education as well as skillful content acquisition. Several programs have been developed over time to be used in the classroom with the aim of improving on the mode of instruction. Program Visualization (PV) tools are an example of such developments with the desire of improving classroom experience during teaching/learning computer programming. Despite that the tools have posted positive results in various universities, teachers seem not to have widely accepted them. This paper seeks to establish the factors that influence the choice of a PV tool for teaching computer programming. The established list of factors indicates that they range from system based issues to other features beyond the system. From the results, it emerges that most teachers are not using emerging contemporary approaches but instead are making use of the traditional approach whose impact is less felt especially for technical courses like programming. These factors form part of taxonomy of PV tools for pedagogy.

Keywords: Pedagogy, Computer Programming, Program Visualization

1. INTRODUCTION

Programming is a course/subject in Computer Science and related fields, and some Engineering disciplines which plays a core role that is applicable in both academic and career developments [1]. Although it "is perceived as a textual discipline, the use of diagrammatic illustration of software has always played a useful role in software development and writing of programs" [2]. Visual techniques are now common in software development, software maintenance as well as teaching the skills thereof. This has therefore seen a significant growth in software visualization (SV). SV can be defined as "the use of typography, graphic design, computer animation and graphics technologies to facilitate the understanding of software" [3]. It provides visual cues, graphical techniques and at times audio means to facilitate human understanding and reasoning [4]. SV has various fields of study, but this paper focuses on Program Visualization (PV). Program Visualization refers to the use of visual cues to illustrate the flow and execution of a computer program. PV tools thus 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 [5].

Over the past three decades, several program visualization (PV) tools have been developed to aid in teaching introductory courses of programming. The trials of these tools in varied universities and other institutions have posted positive results as various students have shown significant improvement of performance [6][7]. Despite the positive results, the tools have not attracted an extensive usage [8] as earlier anticipated despite the mass resources consumed by researchers and developers. The proliferation of these tools may be a contributing factor towards their poor usage because the users have various needs which influence their choice.

This paper is a result of a study that has been ongoing. It elaborates on the factors that influence the teacher's and/or learners' choice of a program visualization tool for use in teaching/learning computer programming. The factors are derivedfrom data collected from students and do not reflect the instructor's opinion. The remaining part of this paper is organized as follows; Section 2 is review of related literature, section 3 explains the methodology that was utilized during the study. Section 4 presents a detailed discussion of the results while section 5 presents the authors' concluding remarks.

2. LITERATURE REVIEW

Learning how to program is a challenging and complex process that requires support of proper educational tools [5]. Research has showed that there is a universal problem in teaching and learning programming courses to most students and teachers [9]. This is clearly evident on novices [10] learning the basics of programming, as well as teaching advanced programming concepts like algorithms, recursion, data structures and object-oriented features to continuing students. This may be attributed to its abstractness [11] and that the students lack concentric model in their everyday life to handle the concepts at hand [9]. Nevertheless, the pedagogical approaches used in teaching the programming courses may be a contributing factor towards the poor performance and understanding of this crucial course in Computer Science and Engineering disciplines. The traditional chalk and blackboard approach seems insufficient to deliver appropriately in classes of computer programming.

To improve on this situation and enforce the understanding of this vital course(s), PV tools have been developed to bridge this gap. Their trial in various institutions has shown that they actually improve the attention of students as well as boosting their interest in the subject hence building a positive attitude. [12] found that students who actively used the JEliot [13] PV tool improved their learning results compared to a control group that did not use it. The results of [14] showed that proper use of PV tools increases the attention and enhances interest of students to the concepts being taught. In their study, [15] assert that usage of VILLE tool, enhanced students' learning regardless of their previous programming experience.

However, despite the number of tools released, not so many teachers and learners have embraced them. It is not because the teachers and students are against their usage, but rather are facing the challenge of choosing the right tool for the right job, and the time required to study and integrate the tool within the syllabus appropriately. Perhaps a proper guideline can assist in choosing the right tool to use [1]. One way to promote the usage of these vital tools is the development of a comprehensive taxonomy that can act as a guideline to the teachers and learners. However, before the development of the classification, the desires of the intended users ought to be collected and documented. This may be utilized to guide the development of such tools and is the sole objective of this paper.

3. METHODOLOGY

This paper is a result of research work that was conducted in the year 2010 through to May 2011 in a Kenyan University (case of Masinde Muliro University of Science and Technology). During the study questionnaires were issued to students who had been introduced to seven different PV tools. The tools were JEliot3, Jeroo, Ville, Alice, BlueJ, Jive and JGrasp. Some of these tools exhibited the characteristics of PV tools while others were integrated development environments (IDEs) with PV features embedded.

4. FINDINGS AND DISCUSSION

Visualization tools have been developed to augment the learning process. These may be in form of teaching aids, toys, models and/or software systems all aimed at enhancing the learning process. However, if such a tool does not meet its objective, it is better for it not to be used at all; and therefore need to be properly used and integrated in during the course work. This is because; it may end up confusing the learners more hence making them loose interest in the course which may be difficult to reassert later.

Out of the total 109 questionnaires issued, 93 were collected back. The questionnaires were verified to ensure no blank questionnaires or having more than 50% questions unanswered formed part of the study. Six (6) questionnaires were not fully answered and had more than 50% unanswered questions hence were considered spoilt. Therefore, out of the 93 questionnaires received back, only 87 were used for the study. This represented close to 80% useful response rate, which was deemed appropriate as this may be a serious disadvantage of this instrument [16].

Before seeking for the factors to be considered to choose a PV tool, respondents were asked their views on the mode of teaching computer programming and whether they considered PV tools as an option.

4.1 Teaching Mode of Computer Programming

Here, the desire was to establish the pedagogical techniques being used in teaching computer programming. It emerged that most Instructors (78%) were using either chalk and board or projectors in the classroom. However, the students felt that there was dire need of other interactive mechanisms other than these two to teach computer programming as shown in Table I.

In your opinion, do you feel that there is need for other teaching mechanism? <Yes/No>

Table I: Respondent's Response on need for other teaching mechanisms

Response	Frequency	%
Yes	83	95.4
No	4	4.6
Total	**87**	**100.0**

From this table, it is evident that the traditional teaching modes are not sufficient enough to deliver desirably in programming classes.

4.2 Usage of Program Visualization

The desire here was to establish the usage of PV tools across the various students' groups. The question:

Have you used any Program Visualization to learn computer programming? <Yes/No>

Table II: PV usage Cross Tabulation per year of Study

	STUDY YEAR			
Response	**First**	**Second**	**Third**	**Fourth**
Yes	75.76%	79.17%	33.3%	40%
No	24.24%	20.83%	66.7%	60%
Total	**100%**	**100%**	**100%**	**100%**

The usage of PVs was observed to be minimal especially in the final last two years of study. Even though 33.3% and 40% of third and fourth year students' respectively responded as having had used PVs in learning computer programming, their assertions were disqualified by the type of examples they gave. Most of them cited having used JCreator which is indeed a Java IDE and neither is a PV tool nor does it have any features of such a tool. On the other hand, the first and second year students cited the usage of BlueJ, Ville and JEliot which were used by the researchers during the study.

During the usage of these tools, observations made and recorded showed that most of them seemed to enjoy using them and that there was some ease in solving various programming exercises. This was further confirmed by the large number of students who came asking for the tools when they were introduced to them.

4.3 Factors influencing choice of tool

During the research, a list of probable factors was listed on a Likert scale of a set comprising Very Relevant, Relevant, Fairly Relevant, Irrelevant and Very Irrelevant. In order to determine which factors were highly considered over others, scores were assigned to each of the Likert scale as shown;

Very Relevant = 2
Relevant = 1
Fairly Relevant = 0
Irrelevant = -1
Very Irrelevant = -2

The weighted score for each factor (metric) was calculated as the sum of the product of frequency of metric and score of relevance, divided by the possible highest score as shown in the following formula;

$$WS = {}^{1}/(N * S_h) \sum (F_m * S_r) \quad \textit{where:}$$

WS is weighted score
N is the total number of respondents used in research
S_h *is the highest possible score*
F_m *is frequency of metric*
S_r *is the score of relevance*

This formula was developed with the assumptions that;

i. The responses are received in form of a Likert Scale

ii. The Likert Scale option are odd in number
iii. The scale is assigned scores which have a neutral point (0) at the middle
iv. The absolute sum of the scores on the left hand side of the neutral point and those on the right hand side is equal

In this research the possible highest score would be two (2) while the total number of respondents being used in this research is eighty seven (87). The results are shown in Table III.

Table III: Factors Weighted Scores

Metric (m)	Very Relevant (2)	Relevant (1)	Fairly Relevant (0)	Irrelevant (-1)	Very Irrelevant (-2)	*Weighted Score (WS)*
AVAILABILITY	37	26	14	3	0	*0.56*
PARADIGM	55	22	4	2	0	*0.75*
USER INTERACTIVITY	29	42	6	2	0	*0.56*
REPRESENTATION	24	40	21	1	0	*0.50*
ANIMATION	29	34	21	3	0	*0.51*
EXTENSIBILITY	31	26	29	1	0	*0.50*
PLATFORM	35	34	8	4	1	*0.56*
META DATA	17	24	17	9	11	*0.16*
PROGRAMMING CONSTRUCTS	36	21	24	5	0	*0.51*
INSTALLATION	31	36	12	1	0	*0.56*
MAINTENANCE	13	30	31	7	1	*0.27*
PEDAGOGY	33	30	19	3	1	*0.52*

For instance the weighted score of maintenance is:

$$WS = {}^{1}/_{(87*2)} \{(13*2) + (30*1) + (31*0) + (7*(-1)) + (1*(-2))\}$$
$$= {}^{1}/_{174} (26+ 30+0+(-7)+(-2))$$
$$= {}^{1}/_{174} (47)$$
$$= 0.27$$

From these values, any metric having a weighted score equal to or more than 0.5 was considered worth. Most of these were used in development of a taxonomy that can be used by both teachers and students to choose a tool for teaching and learning and published in [1]. However, from the respondents' additional response, it emerged that the following factors were highly considered than others;

User Interface

This refers to the display of the computer which allows the human part to interact with the computer itself. This factor was widely mentioned in various questionnaires in different ways such as user interaction and program display.

Simplicity and Ease of Use

A good software system should be simple to understand and easy to interact with. Many respondents expressed that they would strongly consider this especially being novices. The use of colors for easy organization was also mentioned. All These could be summed up by a student's comment that;

> "The first impression a student encounters when they are learning a particular programming language goes a long way in how they view programming as a whole so it should be impressive."

Tool Comprehensibility

This implied how detailed the PV tool is in the various programming constructs and structures. Closely related to this was the support for multi-programming languages to allow a student only to learn a single tool that can be utilized across several other languages.

Availability

This was another factor that was predominantly mentioned by our respondents. Truly, this was such a vital factor because, for maximum utilization of these tools, then they ought to be readily available to the intended users. Others which were together grouped in this factor were cost and affordability.

Extensibility
This is a feature that allows one to add more functionality to the existing ones of a PV tool.

Platform
This was widely mentioned as operating system or portability. This implied the OS platform in which the PV tool can run. Some could be cross-platform while others were specific to certain operating systems.
Others which were lightly mentioned were code completion and generation, memory utilization, speed of operation, durability and animations.

5. CONCLUSION

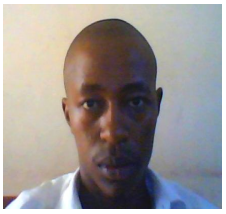
Program Visualization tools are a worthy way of teaching computer programming. They improve the student's attention and enhance their interest in the course. Nevertheless, for this to be fully achieved, teachers and students need to know exactly what they are looking for in any pedagogical tool. Improper choice and use of teaching aids can lead to enhancement of negative attitude towards a subject which may go a long way before being overcome. The main features that may stand out for a PV tool are therefore the tool's user interactivity, its extensibility and availability. Further, a PV tool should be compatible with the platform that is being used in delivery of the course. These and others as listed in this paper if well utilized will significantly aid the users in selecting a tool for use thus in return improved scores and acquisition of skill in computer programming.

REFERENCES

[1] Mutua Stephen, Wabwoba Franklin, Ogao Patrick, Anselmo Peter and Abenga Elizabeth, "Classifying Program Visualization Tools to Facilitate Informed Choices: Teaching and Learning Computer Programming", (International Journal of Computer Science and Telecommunications), Volume 3, Issue 2, February 2012, pp. 42 – 48
[2] Calum, A., "Software Visualization in Prolog", Dissertation submitted for the degree of Doctor of Philosophy in Queens' College, (Cambridge December 1999)
[3] Guest Editors' Foreword, "Software Visualization", (Journal of Visual Languages and Computing) Vol. 13, pp.257 – 258, 2002
[4] Petre, M., "Mental imagery and software visualization in high-performance software development teams", (Journal of Visual Languages and Computing) Vol. 21, pp. 171–183, 2010
[5] Bednarick, R., Moreno, A., Myller, N., & E, Sutinen, "Smart Program Visualization Technologies: Planning a Next Step", Proceedings of the Fifth IEEE International Conference on Advanced Learning Technologies (ICALT'05), 2005
[6] Kouznetsova S, (2007), Using Bluej and Blackjack to Teach Object-oriented Design Concepts In Cs, (Journal of Consortium for Computing Sciences in Colleges, pp. 49 -55, April 2007
[7] Kasurinen, J., Mika, P. & Uolevi, N., "A Study of Visualization in Introductory Programming", In proceedings of the PPIG conference, Lancaster 2008
[8] Bassat, L. & Mordechai, B., "We Work So Hard and They Don't Use It: Acceptance of Software Tools by Teachers", Proceedings of the ITiCSE' Working group, Dundee, Scotland, June 23–27, 2007
[9] Robins A, Rountree, J. & Rountree, N., "Learning and teaching programming: A review and Discussion", (Journal of Computer Science Education), Vol. 13-No. 2, pp. 137–172, 2003
[10] Sanders, D & B. Dorn, "Using JEROO to Introduce Object-Oriented Programming", 33rd ASEE/IEEE Frontiers in Education Conference, November 5-8, 2003
[11] Lahtinen, S. P, Sutinen, E. & J. Tarhio, "Automated Animation of Algorithms with Eliot", (Journal of Visual Languages and Computing), Vol. 9 Issue 3, pp.337–349, 1998
[12] Bassat Levy, R., Ben-Ari, M., & Uronen, P. A., "The Jeliot Program Animation System", (Journal of Computing Education),Vol. 1, pp. 15–21, 2000
[13] Moreno, A, Myller N, & Sutinen, E., "Visualizing Programs with Jeliot", In the proceedings of The Association of Computing Machinery, May 25-28, 2004
[14] Ebel, G. & Ben-Ari, M., "Affective effects of program visualization", In Proceedings of the 2nd International Computing Education Research Workshop (ICER'06), pp. 1-5, 2006

[15] Rajala, T., L. Mikko-Jussi, K. Erkki, Salakoski, P., "VILLE – A Language-Independent Program Visualization Tool", Proceedings of the Seventh Baltic Sea Conference on Computing Education Research, Koli National Park, Finland, Vol. 88 November 15- 18, 2007

[16] Kombo, D. K & Tromp, D. L. A, Proposal and Thesis Writing: An Introduction, (Daughters of St. Paul; Nairobi), 2006



Mr. Mutua Stephen received his B.Sc. in Computer Science (First Class) in 2008 from Masinde Muliro University. He has successfully completed his Master's studies. He has more than three years teaching experience and his research interests are in Software Visualization, Mobile Computing and Use of Technology in Pedagogy. He is an upcoming scholar who has published widely in various journals. He is currently working lecturing in Masinde Muliro University, a position he was awarded following his excellence performance in his first degree.

Dr. Elizabeth Abenga received her PhD degree from Moi University, Kenya in 2005. She is a Curriculum and Educational Technology Specialist. Her research interests are in Pedagogy, Teacher Education and Emerging Instructional Technology. She is a member of Organisation of Social Science Research in Eastern and Southern Africa (OSREA) Kenya Chapter, Association for the Advancement of Computing in Education. (AACE), Association of Third World Studies (ATWS). Regional African Alumni UNISTAFF Network and Kenya DAAD Scholars Association. She is currently working as a senior lecturer in Department of Curriculum and Instructional Technology Masinde Muliro University of Science and Techno logy, and Director of International Relations and Academic Linkages. She has a teaching experience of 24 years.

Franklin Wabwoba is a lecturer in Information Technology and Chairman of Department at Kibabii University College, Kenya. He holds a Master of Science (Computer Applications) degree from Kenyatta University, 2007; Endorsement (Educational Management) from University of South Africa, 1997 and Bachelor of Education (science: Mathematics and Computer Science) degree from Egerton University, 1991 and is currently a PhD (Information Technology) candidate at Masinde Muliro University of Science and Technology. He is a member of Association of Computing Machinery. He has ICT industrial experience having worked with Mumias Sugar Company. He has a number of publications in peer reviewed journals. He has presented several papers in scientific conferences. He has a strong research interest in green ICT, the impact of ICT applications on the community, and integration of ICT into education.

Prof. Patrick Ogao received his PhD from University of Utrecht, Utrecht (The Netherlands. He is a specialist in Geographical Information Systems (GIS) and Visualization of Geo Spatial Data He is currently the dean of the School of Computing & Information Technology in The Kenya Polytechnic University College. He is a member of the ACM SIGGRAPH group. He has published widely and has research interests in Software Visualization, Geographic Information Systems (GIS) and Bio-Informatics. He has over 15 years of teaching experience.