ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 8, Issue 11, November 2019

Data Quantification to Measure Effectiveness of an Interactive Elementary Programming Model

Payal Gohel, Dr. Kishor Aatkotia

Researcher Scholar, Saurashtra University, Rajkot, India

Professor, Saurashtra University, Rajkot, India

ABSTRACT: As the Indian economy launched its ambition to achieve India's target of 5 trillion economy by 2024, high skilled labor force is going to be necessity of the hour. India being the youngest country in the world, it has to use leverage of its top position in computing and software services. In computerscience, programming is considered to be the most challenging subject to learn and understand especially for new students. An extensive investigation of past researches was conducted and identified the underlying challenges of this issue and how to resolve have been a focused area for the research community. Even with the advancement of technology, education sector is still not fully utilizing the power of it. Therefore, it's time that appropriate teaching methodology should be developed using interactive teaching curriculum to ease students' learning difficulty caused by lack of experience by teachers, complicated interfaces and necessary base understanding for a programmer. Amongst many aspects of a methodology, author presented a framework in the realms of technological, theoretical and pedagogical concepts. In addition, quantitative methods was applied to identify the gaps and challenges for beginners. An extensive survey targeting 360 people was conducted which helped to identify the trends of current state in programming and what can be done to address it. Based on literature and data collection findings, this research proposes a theoretical process framework to design and develop Visualized ProgrammingEnvironment for new students. Then, it presents the statistical assessment of the proposed research model conducted through quasi-experimental designs to understand the impact of it. After the rigorous data quantification, a significant improvement was observed in VProEn users' cognitive abilities and ease of learning programming.

KEYWORDS: Smart education framework, Prototyping, Theory of constructivism, Cognitive Load Theory, Visualization methodology, Regression Analysis.

I. INTRODUCTION

Learning a programing at the college level becomes an entry point for most professionals. However, many researchers have identified that student are facing learning difficulties in the conventional way of teaching. There are various reasons cited throughout many researchesincluding they don't understand the teacher; they are hesitant to ask any questions; subject is very complicated to learn; it's very boring to sit through entire class and read programming syntax. Even with the advancement of technology, education sector is still not fully utilising the technology. Therefore, it's high time that appropriate learning and teaching methodologies should be developed using interactive teaching mechanism to ease students' learning difficulty, caused by lack of experience and necessary understanding.

The author argued, part of a problem has been a tendency to only look at the technology and not how it is used. Merely introducing technology to the educational process is not enough. The question of what teachers need to know and how students' needs to learn in order to appropriately incorporate technology into their teaching has received a great deal of attention in past decade [1] [2]. Use of Multimedia in a teaching environment, in particular with the integration of cognitive load theory has the potential to enhance the student's 'attention span and enable focused learning. Such methodology has shown tremendous potential for students which help them to guide in the study, minimize the difficulty, and further cut down the mental load. Therefore, The primary aim of the proposed research is (a) the critical

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 8, Issue 11, November 2019

investigation of different learning theories to outline main principles for coding tool for beginners, (b) the design and development of a novel coding model to demonstrate identified design principles and (c) the verification and validation of the proposed model to assess the enhancement of the learning for young students.. In this paper, the author focuses on the first objective to identify the critical design principles for the effectiveness of the tool to enhance cognitive learning.

II. CRITICAL INVESTIGATION OF LEARNING THEORIES

In the context of this research, learning theories are conceptual frameworks describing how the coding skill is acquired, understood, and maintained during learning stage. From various learning theories, the propose research focuses on two critical theories, i) Constructivism theory and, ii) Cognitive Load theory. In-depth investigation of this theories were conducted in the early publication [9].

2.1 REVIEW OF CONSTRUCTIVISM THEORY

Constructivism is a theory of knowledge and how can it be achieved by people. Many research studies have referred to constructivism in educational tool. However, one of the first research to study of the implications of applying constructivism was conducted by [3][4]. The author identified the learning difficulties in students when they used a what-you-see-is- what-you-get (WYSIWYG) word processor. The author found that, CS students lacked a cognitive framework to guide them, in order to gain focussed knowledge from their regular interaction with a computer. Second the computer presents an accessible ontological reality. A number of researchers have focused on this area. The InSTEP [5] was developed to provide a constructivist learning experience for computer engineering students as an introductory course. [6] developed a constructivist approach in creating teaching material and guidelines for basic coding classes. He echoed that constructivism theory enhanced students' understanding of the subject material. Then, A pedagogical approach based on a constructivism was presented by [7] for teaching object-oriented concepts for students.They conducted three case studies on how real-life data can be used from constructivism to teach the sorting algorithms, solve puzzles and recognize groups from their multiplication tables. [8] applied constructivism to demonstrate the concept of 'static' in Java program and why it is often hard to understand.

2.2 REVIEW OF COGNITIVE LOAD THEORY

This theory aims to create a conceptual framework of how information is understood intellectually by an individual to achieve greater learning outcomes. Many researchers have undertaken this area of research in teaching tools. [11] presented a 4C/ID model for developing instructional programs for complex skills acquisition. [12][13] presented how cognitive load theory can assist multimedia learning and the design of such software. One of the experiments was conducted using text and then, images and text both at the same time. [14] developed a pedagogical design using cognitive load theory and other theories for teaching Object Oriented Programming concepts. [15] investigated the effect of various strategies on the different learning measurements for cognitive load theory to acquire programming-knowledge especially loops acquired. [16] presented case-study for particular programming concepts. [17]The model was developed using the Cognitive load and Human computer interaction principles. Their review showed that there is commonality between aforementioned principles, namely the reduction of unnecessary load in users' mind. Many years of research in the field of cognitive load theory in various disciplines have been undertaken where researchers have demonstrated some important techniques that minimise cognitive load. These techniques are: the modality, the worked-example, the expertise reversal, the redundancy,the goal-free, the split-attention effect and, the completion problem effect [18][19][20][21]. However, not every researcher found cognitive load theory useful in enhancing learning [22][23][24], there have been some criticism. [25] raised some concerns regarding the effectiveness of cognitive load theory theory in practical environment. A number of researchers studied the results that contradicted cognitive load theory's predictions and identified some critical methodical problems.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 8, Issue 11, November 2019

III. SURVEY METHODOLOGY

The survey was sent by e-mail, posted on social network and group messaging to a statistical sample of 360 people. This sample of group was split in four categories: academic researchers, teachers, computing students, and programmers. The prior consent was taken, and the answers provided were kept confidential and used for study purposes.

3.1 Quantitative results and discussions

Most conducted survey indicates, the return rates for Internal surveys will generally receive a 30-40% response rate on average, compared to an average 10-15% response rate for external surveys. It also depends on the selected demographics, enthusiasm towards the topic, focused participations etc; The findings presented here are based on an overall 34.6% return rate, 194 respondents. Even though an average response was obtained, the findings of the survey presented a clear trend towards the current programming tools and its complexity while learning for beginners, useful information about the respondent's perspective and addressed specific concerns while learning programming. The distribution of responses with respect to the three categories of targeted population is shown in Figure 2. As it shows, students participated overwhelmingly amounting to 70.6% that gave a clear trend of their difficulties and challenges in learning programming in existing teaching methodologies. There were also 22.2% programmers who gave us some interesting insight about present IDEs and what can be done to improve the current state. Among those respondents, more than 58% population were exposed to programming between 1-3 years that shows most of the respondents were either familiar with the programming or had some knowledge about it.

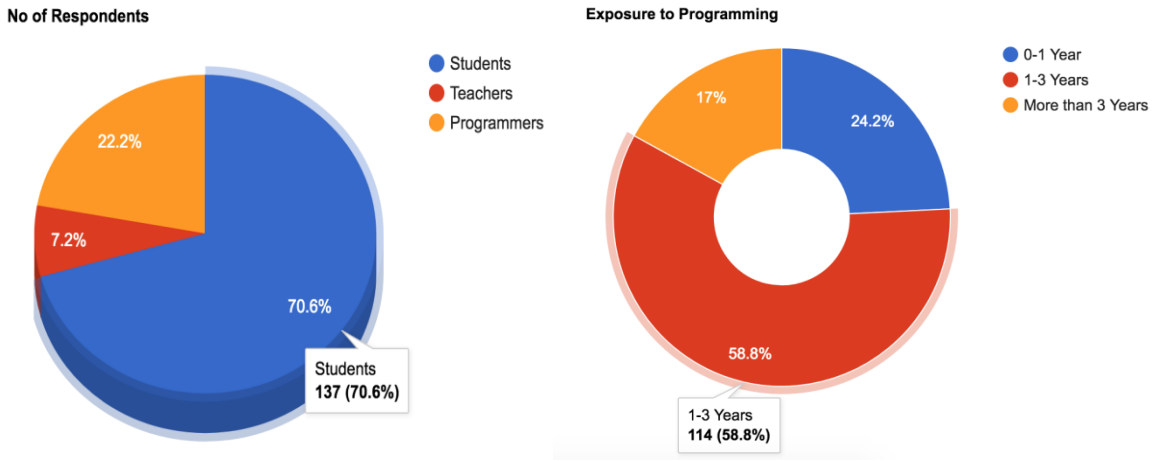

Figure 1 : Questionnaire Population Classification

Next chart spells out the languages and environment they were exposed for the first time in their life. This says something about today's teaching curriculum that are exposing the students to complex programming structure without building their base and creating a basic understanding of natural programming language. Majority of the respondent enjoyed programming as a subject, almost 89% but found it boring while learning in class (figure 3).

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 8, Issue 11, November 2019

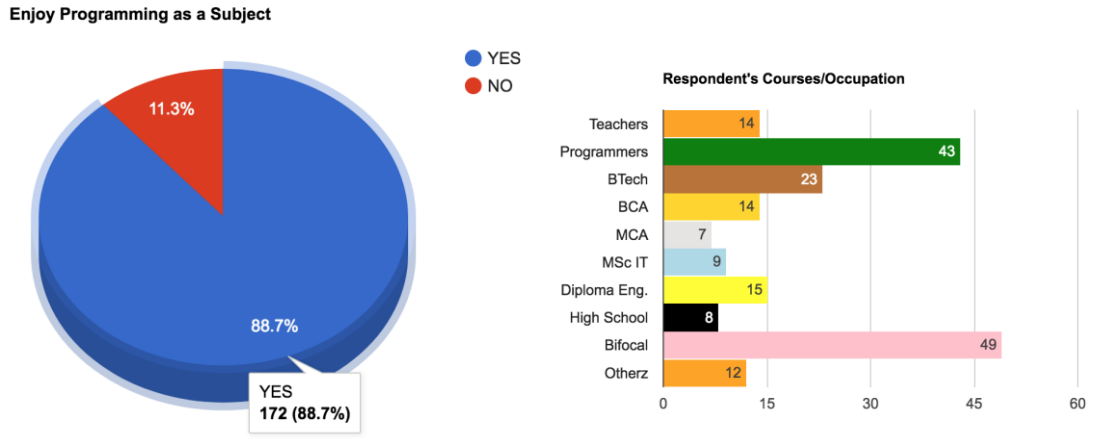

Figure 2 : i) Programming as a Subject, and ii) Respondent's domain

While selecting the population, authors decided to select students from School to diploma, bachelors and master to teachers and programmers. It gave a wider perspective about how they see programming as a subject and how to develop an effective methodology for beginners. Surprisingly maximum respondent cited part of a curriculum as their reason for learning programming, some said they are using particular IDE because their teachers said so while some minority suggested they are learning only for exams (figure 4).

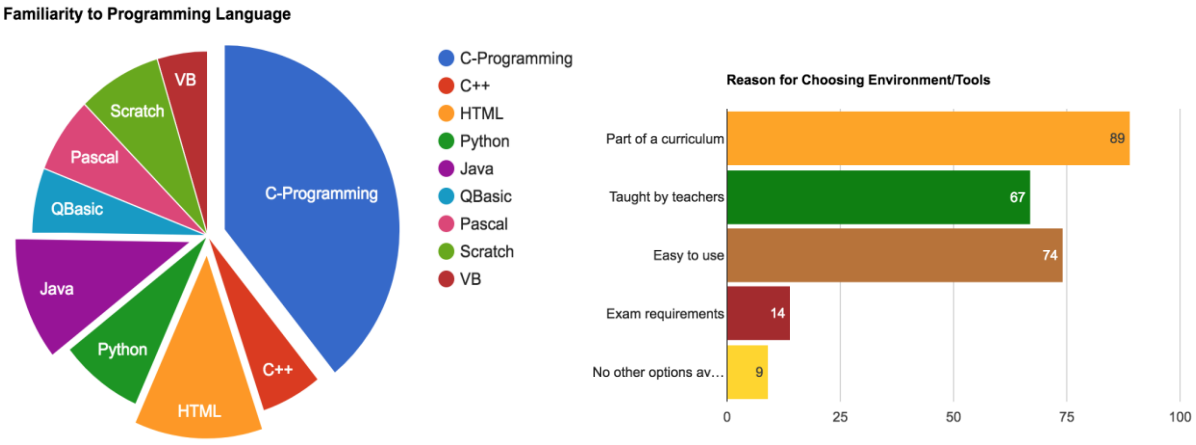

Figure 3 : i) Exposure to Programming and, ii) Reason for choosing Environment/Tools

However, there were majority of the respondent who enjoyed programming as a subject and found Interactive UI and Easy declaration one of the important features for using any specific PL/IDE. One of the major revelations was, more than 60% people found syntax semantics in different environment confusing including complex IDE and dull interface (figure 5). They thought aforementioned are the main disadvantages.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 8, Issue 11, November 2019

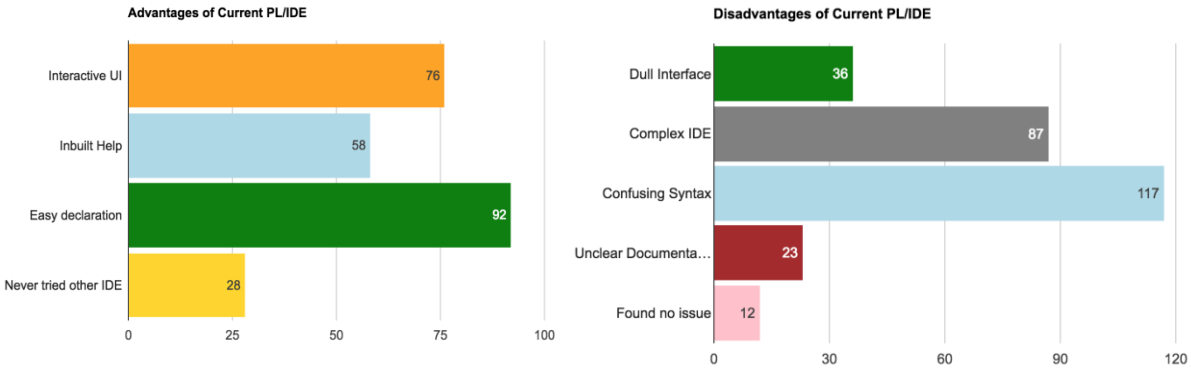

Figure 4 : Advantages and Disadvantage of Current PL/IDE

When the author asked for desired features for proposed PL/IDE, majority suggested to have natural language easy commands, step by step instruction to write program and visual interface and some desired intelligent hints while writing codes (refer to figure 6). When the author asked about their current mode of teaching majority answered, they are learning in traditional classroom setup or in labs or learning from friends. Majority of the programmers said they prefer to learn from video tutorials or getting help from online communities. They were quite uptodate with their knowledge and wanted instantaneous help.

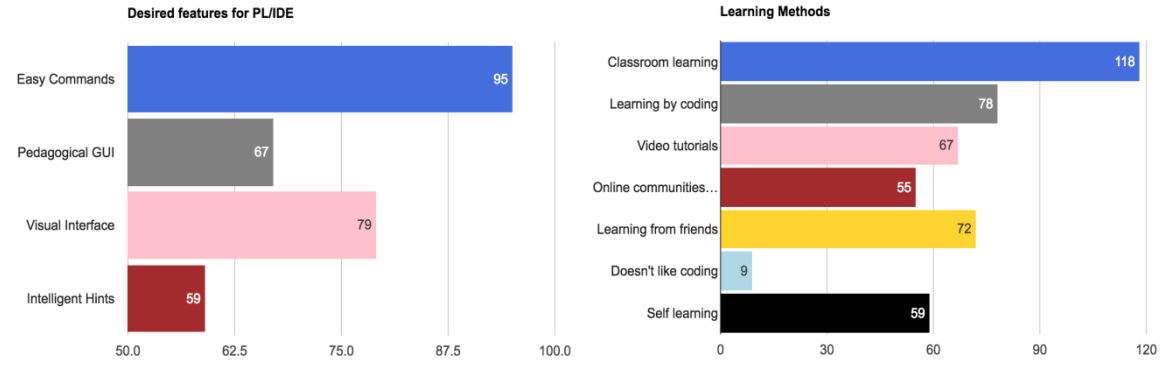

Figure 5 : i) Desired features of the PL/IDE and ii) Learning Methods for programming students

In another question, respondents addressed their challenges in current course. Different syntax for different programming environment was major complain from almost 35% respondent while 21% opined learning in classroom is the most boring way to learn programming. There were 15% who said, they don't understand teacher at all. Most beginners have difficulty in learning a programming in C and Java environment especially when they are beginning to learn (refer to figure 7). Most students found difficulty in basic understanding of difference between data types and variable. Hence, having a instruction in natural language for the beginners can be easy to understand and program. Many participants said they knew the entire logic and steps for the program but still they couldn't create the program. They were missing a basic understanding to code.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 8, Issue 11, November 2019

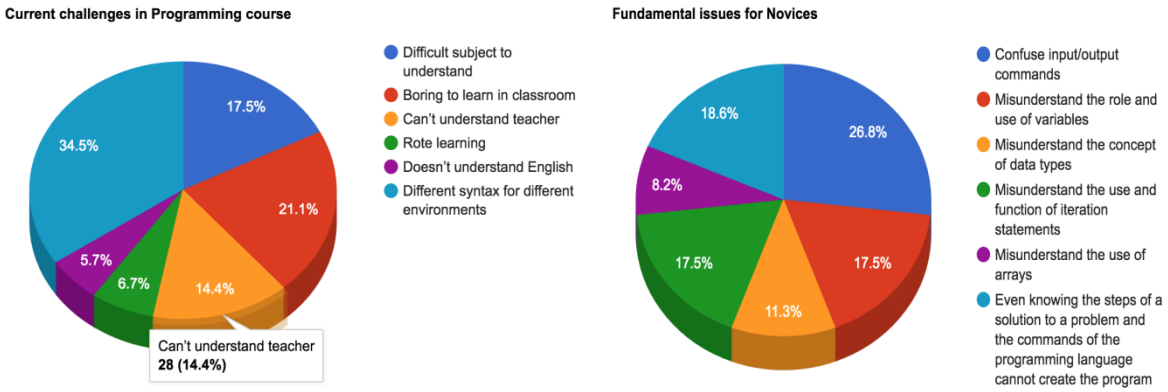

Figure 6 : i) Current Challenges in Programming courses and ii) Fundamental issues for novices

When the author asked for important features they would like to have in proposed PL/IDE, majority responses were simplified syntax, instant error detection and visual way of coding (refer to figure 8). It seems they were quite clear about what they would prefer in a programming environment. Almost unanimously all participant agreed that a Icon based visual programming is a better way to get exposed to programming as it is cognitive and constructive way to learn it.

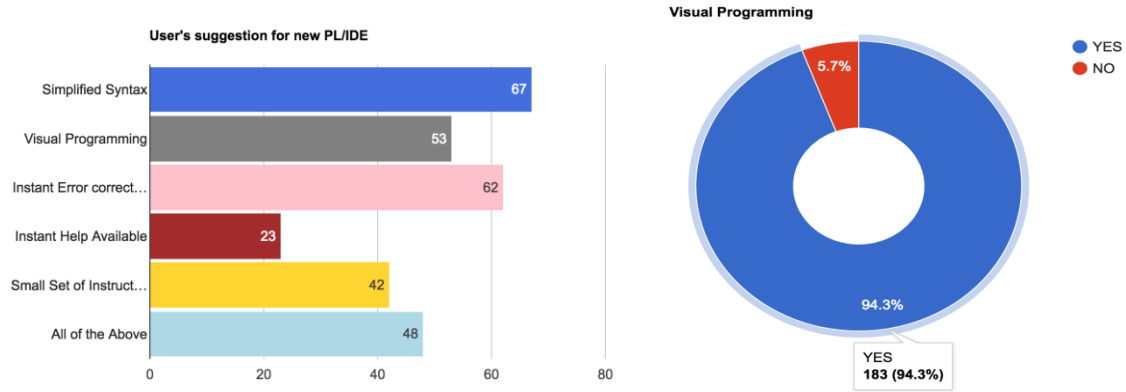

Figure 7 : i) Respondent's suggestion for new PL/IDE and, ii) User's preference for Visual Programming

5. PROPOSED THEORETICAL PROCESS FRAMEWORK

The basis of our framework is the understanding that learning and teaching is a highly complex activity that draws on many kinds of knowledge [31]. In this particular context, the implications of developing a framework go beyond a coherent way of thinking about learning theories, multimedia and pedagogy integration. Many scholars have argued that knowledge about technology cannot be treated as context-free and that good methodology requires an understanding of how technology relates to the pedagogy and theoretical contents [32][33][34].Authors argued that a conceptually based theoretical framework about the relationship between technology and pedagogical learning can transform the conceptualization and the practice of learning and teaching. It can also have a significant impact on the kinds of research questions that we explore. How this framework has guided our research and analysis of the effectiveness of our pedagogical approach.An additional objective of this work to offer a theoretical framework that integrates the pragmatic, technological and the theoretical components together. The discussion of these learning theories and the identification of critical learning challenges of computing students in their programming subject via

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 8, Issue 11, November 2019

quantitative and qualitative survey provided the basis to develop a conceptual process framework. Figure 10 present the top level process framework.

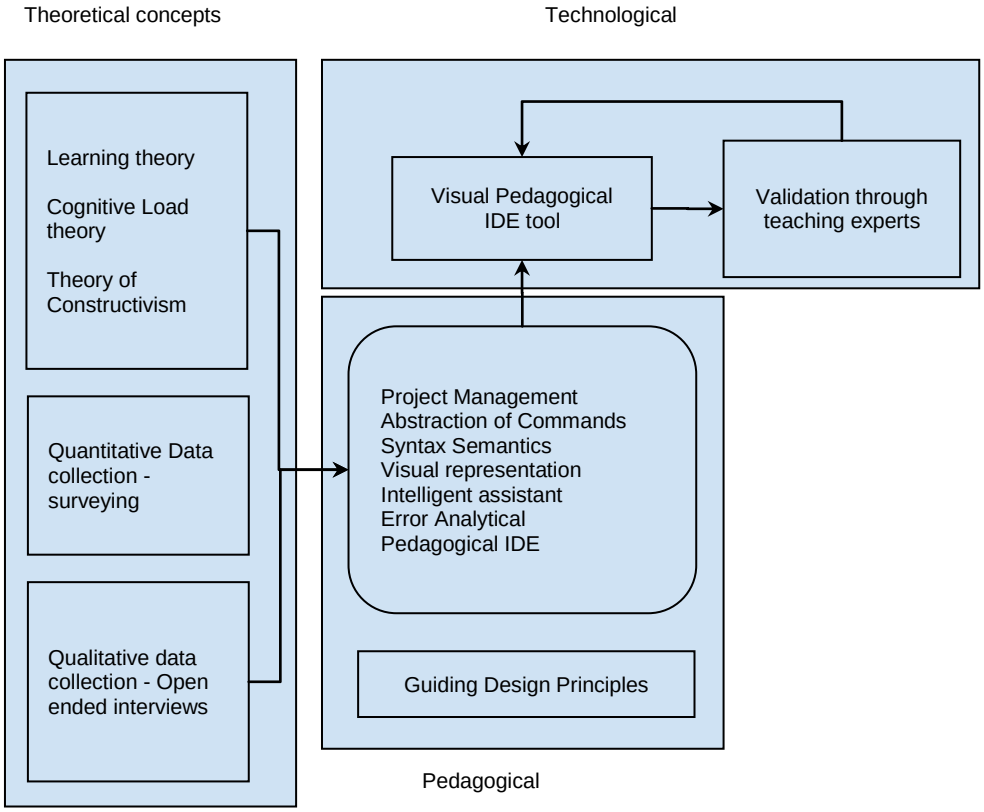

Figure 8 : Top level Process framework

VI. EVALUATION OF THE VproEn Model

VProEn (Visualised Programming Environment) was designed as a coding interface adheringall the above-mentioned guiding principles (table 1). So, VProEn and Turbo-C were chosen to represent the Independent Variable (IV) that will comply with the proposed key features. Turbo C is a coding interface that is used for teaching introductory programming language for computing students. It was the first computing language many students were exposed to as per our survey questionnaire. The Dependent variable (DV) was about the coding skill level of new learners. Then, the quality of learners' cognitive models was the mediator (M) [35]. The evaluation and assessment of the VProEn coding interface as an internal parameter allowed, i) the proposed model validation, ii) the impact of such research on acquisition of a coding and level of skills quality on 21st century learners and, iii) The comparative study of proposed design model and partially complied tool. The proposed research conducted a quasi-experimental study to evaluation the efficacy of the model.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 8, Issue 11, November 2019

Table 1 : Key design for the two coding interfaces used in the Quasi-Experimental Designs

	Natural Language IDE	Syntax and Semantics	Visualization	Level of Abstraction	Set of Commands	Provision of Error messages	Interactive IDE
Turbo C	X	X	X	√	X	X	X
VProEn	√	√	√	√	√	√	√

In the experiment, the students were divided into P1, P2 and P3 groups sizing approximately 20 students each. To conduct the experiment in class, all groups were asked to use Turbo C and the VProEn modelling as their coding interface. The author delegated the teaching material, classroom examples, homework and teaching herself with the help of college. Moreover, the author was actively involved in the process and monitored all groups to gather unbiased data throughout the experiment session. The coursework to learn programming included First Theoretical notes about different coding concepts, second illustration of coding concepts in coding interface, Third Programming examples for learning to apply in coding concepts and Fourth, An exercise. In each coursework, students have option to learn coding either using a nature language or the commands that were provided in the coding interface. First few commands were provided to them to encourage and made them interested in learning by giving easy exercise. The author had to keep in mind that this was the first time that all his target population were learning to code. At the same time, they may not have access to a software at their home. So, their productivity and focus were very crucial. At the end of each session, the author asked to return all the assignments and made sure that they finish their exercise in the classroom itself. There were two reason for it, first student can finish their coursework while they are learning and second their use of it to learn while its fresh.

After collecting the data from the designed experiment, data quantification was carried out. The important objective for quantifying the data was to verify the mediation analysis. Therefore, the results gathered from quantifying the data were to identify: i) Coding interface can have impact on participant's skill, ii) Changes in the significant percentage of variance. The author has decided to use following test that can present the results are: (a) analysis of variance (ANOVA), and (b) analysis of covariance (ANCOVA) and, (c) hierarchical multiple regression (HMR) when covariates should be included. The only common significant predictor in the three quantified data was Participant's lack of experience. when the coding interface was included in the regression model of each quantification, all covariates that were significant before the inclusion ceased to be, with the exception of Participant's lack of experience.

Table 2 : Regression Analysis of imperative-coding

internal parameter	external parameter	Statistical test	Covariates
Coding interface	Scores of the first imperative knowledge test	HMR	lack of Computing experience

The mediation quantification establishes whether the independent variable (IP) has significant impacts on the dependent variable (DP). The skill level of imperative cognizance was quantified using a number of parameters that was gathered from data collection from participants ability to predict the outcome of codes, complete missing codes

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 8, Issue 11, November 2019

and change existing programs. Before testing user's ability, all the concepts were taught to them and they were already aware of each of this command. The difference in the group scores between VProEn and Turbo C was small to medium.

Table 3 : Descriptive Statistics of the sample data quantification

Group	n	Mean	SD	SE
VProEn	35	10.34	3.58	0.76
Turbo C	27	5.27	5.41	1.18

Note: Because not all students attended the test session, sample size was reduced in VProEn group
The outcomes of statistical assessment of collected data revealed that, at the end of the study, first timer coding learners who used VProEn tool fulfilling proposed design concepts seems to have augmented neither a higher level of coding skills nor richer logical models in introductory coding than first timer coders who used Turbo C fulfilling partial set of design concepts.

VI. CONCLUSION AND FUTURE WORK

Many researchers suggested that learning programing for novices has always been demanding and frustrating. There are various reasons identified throughout many researches including sometimes they don't understand the teacher; they are hesitant to ask any questions; subject is very complicated to learn; it's very boring to sit through entire class. Therefore, to investigate such issues, this paper conducted an extensive review of two important learning theories that have a greater impact on academic learning in computer programming, i) theory of constructivism and, ii) Constructive Load Theory. From the review of the past literature, some vital gaps in knowledge are identified. First, many researchers have employed constructivism and cognitive load theory into their research, but this research proposes to integrate both the learning theories for the design of educational programming tool for beginner students. Second, many researchers discussed the set of design principles using aforementioned theories but not many researchers have measured the impact of such educational programming tool empirically. Then, to gather quantitative data survey was conducted, targeting 560 population. Based on the quantitative data collection and critical investigation of no of research, this paper proposes a theoretical process framework of a novel methodology as a validation of these proposed principles. Data quantification of the test scores indicated that VProEn users appeared to perform better than users of Turbo C in the following coding aspects: i) Declaration of variables and use of i/p and o/p commands, ii) Use of correct semantics, iii) Identify the correct order and scope of the code, iv) Executability of the code and correctness of program execution. In nutshell, a significant improvement was observed in VProEn users than Turbo C users earlier but in the longer run no significant evidence was found.

REFERENCES

1. International Society for Technology in Education. (2000). National educational technology standards for students: Connecting curriculum and technology. Eugene;
2. Zhao, Y. (Ed.). (2003). What teachers should know about technology: Perspectives and practices. Greenwich, CT: Information Age.
3. Gonzalez, G. (2004). Constructivism in an introduction to programming course.Journal of Computing Sciences in Colleges, 19(4), 299-305
4. Mannila, L. & de Raadt, M. (2006). An objective comparison of languages for teaching introductory programming. 6th Baltic Sea conference on Computing education research: Koli Calling 2006, Uppsala, Sweden.
5. Ben-Ari, M. (1998). Constructivism in computer science education. ACM SIGCSE Bulletin 30(1), 257-261.
6. Ben-Ari, M. (2001). Constructivism in computer science education. Journal of Computers in Mathematics and Science Teaching, 20(1), 45-73.
7. Odekirk-Hash, E. & Zachary, J. L. (2001). Automated feedback on programs means students need less help from teachers. SIGCSE Bulletin, 33(1), 55-59.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 8, Issue 11, November 2019

8. Hadjerrouit, S. (2005). Constructivism as guiding philosophy for software engineering education. SIGCSE Bulletin, 37(4), 45-49.
9. Beynon, M. (2009). Constructivist computer science education reconstructed.Innovation in Teaching And Learning in Information and Computer Sciences,8(2), 73-90.
10. Milner, W. W. (2010). Concept development in novice programmers learning Java. PhD thesis, The University of Birmingham, Birmingham.
11. Lee, Y.-J. (2011). Empowering teachers to create educational software: A constructivist approach utilizing etoys, pair programming and cognitive apprenticeship. Computers & Education, 56(2), 527-538
12. Van Merriënboer, J. J. G. (1990a). Instructional strategies for teaching computer programming: Interactions with the cognitive style reflection-impulsivity. Journal of Research on Computing in Education, 23(1), 45-53.
13. Van Merriënboer, J. J. G. & Krammer, H. P. M. (1987). Instructional strategies and tactics for the design of introductory computer programming courses in high school. Instructional Science, 16(3), 251 -285.
14. van Merriënboer, J. J. G. & Paas, F. (1990). Automation and schema acquisition in learning elementary computer programming: Implications for the design of practice. Computers in Human Behavior, 6(3), 273-289
15. Van Merriënboer, J. J. G. & Kirschner, P. A. (2007). Ten steps to complex learning: A systematic approach to four-component instructional design. Mahaw, New Jersey: Erlbaum.
16. Garner, S. (2002a). Colors for programming: A system to support the learning of programming. Informing Science & IT Education Conference (InSITE), Cork, Ireland.
17. Caspersen, M. E. & Bennedsen, J. (2007). Instructional design of a programming course: A learning theoretic approach. 3rd international workshop on Computing education research, Atlanta, Georgia, USA.
18. Gray, S., Clair, C. S., James, R. & Mead, J. (2007). Suggestions for graduated exposure to programming concepts using fading worked examples. 3rd international workshop on Computing education research, Atlanta, Georgia, USA.
19. Hollender, N., Hofmann, C., Deneke, M. & Schmitz, B. (2010). Integrating cognitive load theory and concepts of human-computer interaction. Computers in Human Behavior, 26(6), 1278-1288.
20. Abdul-Rahman, S.-S. & du Boulay, B. (2014). Learning programming via worked examples: Relation of learning styles to cognitive load. Computers in Human Behavior, 30(0), 286-298.
21. Chong, T. S. (2005). Recent advances in cognitive load theory research:Implications for instructional designers. Malaysian Online Journal of Instructional Technology, 2(3), 106-117.
22. Sweller, J., van Merriënboer, J. J. G. & Paas, F. (1998). Cognitive architecture and instructional design. Educational Psychology Review 10(3), 251 -296.
23. van Merriënboer, J. J. G. & Sweller, J. (2005). Cognitive load theory and complex learning: Recent developments and future directions. Educational Psychology Review, 17(2), 147-178.
24. van Mierlo, C., Jarodzka, H., Kirschner, F. & Kirschner, P. A. (2011). Cognitive load theory and e-learning. In Z. Yan (Ed.), Encyclopedia of Cyberbehavior: IGI Global.
25. Ayres, P. & van Gog, T. (2009). State of the art research into cognitive load theory. Computers in Human Behavior, 25(2), 253-257. Baddeley, A. (1992). Working memory.
26. Paas, F., Van Gog, T. & Sweller, J. (2010). Cognitive load theory: New conceptualizations,specifications, and integrated research perspectives. Educational Psychology Review, 22(2), 115-121.
27. Verhoeven, L., Schnotz, W. & Paas, F. (2009). Cognitive load in interactive knowledge construction. Learning and Instruction, 19(5), 369-375.
28. Moreno, R. (2006). When worked examples don't work: Is cognitive load theory at an impasse? Learning and Instruction, 16(2), 170-181.
29. Schnotz, W. & Kürschner, C. (2007). A reconsideration of cognitive load theory. Educational Psychology Review, 19(4), 496-508.
30. Gomes, A. & Mendes, A. J. (2007). Learning to program - difficulties and solutions. International Conference on Engineering Education – ICEE 2007, Coimbra, Portugal.
31. Rajiv Chavada, Nashwan Dawood, Mohamad Kassem (2012). Construction workspace management: the development and application of a novel nD planning approach and tool, ITcon Vol. 17, pg. 213-236, http://www.itcon.org/2012/13
32. Hughes, J. (2005). The role of teacher knowledge and learning experiences in forming technology-integrated pedagogy. Journal of Technology and Teacher Education, 13(2), 277–302.
33. Lundeberg, M. A., Bergland, M., Klyczek, K., & Hoffman, D. (2003). Using action research to develop preservice teachers' beliefs, knowledge and confidence about technology [Electronic version]. Journal of Interactive Online Learning, 1(4). Retrieved June 29, 2004, from http://ncolr.uidaho.com/jiol/archives/2003/spring/toc.asp
34. Neiss, M. L. (2005). Preparing teachers to teach science and mathematics with technology: Developing a technology pedagogical content knowledge. Teaching and Teacher Education, 21(5), 509–523.
35. MacKinnon, D. P. (2008). Introduction to statistical mediation analysis. New York: Erlbaum.