iBaTs: Interactive Bash Shell Adaptive Tutoring System

Narasimha Shashidhar karpoor@shsu.edu
Department of Computer Science
Sam Houston State University
Huntsville, TX 77384, USA

Cihan Varol cvarol@shsu.edu
Department of Computer Science
Sam Houston State University
Huntsville, TX 77384, USA

Keerthi Koneru keerthi@shsu.edu
Department of Computer Science
Sam Houston State University
Huntsville, TX 77384, USA

Abstract

A vast majority of students in computing and related disciplines expect to interact with their systems and computing devices using a graphical user interface. Any other means of interacting with a device is deemed unseemly and is quickly met with frustration and rejection. This can partly be attributed to the fact that most operating systems and the tools that run on these platforms offer a rich "point-and-click" interface in an effort to make their systems user friendly. However, in contrast, when it comes to the study of system and cyber security, a mastery over the console and the command-line interface is imperative. In our experience in teaching most courses on system and cyber security, students seem to have the greatest difficulty in using the console/command-prompt/shell. This issue is further exacerbated since many security and related open source forensics tools are designed to run in a Unix-based environment, typically a shell, and even fewer students are familiar with the UNIX environment and find the entire experience all the more daunting. Even the simple command-prompt, ubiquitous on all Microsoft Windows operating systems, is met with significant disdain by today's students, both at the graduate and undergraduate levels. There are several solutions that have been proposed and designed to alleviate this exact issue in the field of computer programming. Video tool, Dragon Drop Pictorial Programming, Alice and Jpie are various stand-alone tools introduced to ease the inherent challenges in learning a new programming language and environment. To alleviate this situation, in this paper, we propose the first tool of its kind, to the best of our knowledge, which aims to tutor a console application using a graphical interface and adapts to the students' progress. The ultimate aim is to eliminate students' dependence on graphical interfaces and convert her to a power user of a system. Our tool, called *Interactive Bash Shell Adaptive Tutoring System (iBaTs)*, enables students to familiarize themselves with the UNIX environment and the Bash Shell on a Windows operating system. In this work, we discuss the architecture of our tutoring program and demonstrate that our system sports several innovative pedagogical features that makes it a unique, fun, encouraging and adaptive learning environment. To the best of our knowledge, this is the first such effort that aims to address this issue.

Keywords: Adaptive Learning, Bash, Console, Command-line, Interactive Shell, Shell Scripting.

1. INTRODUCTION

One of the main challenges that engineering and STEM related academic programs in the United States are facing is the recruitment and retention of female and minority students in their programs. According to the CRA Taulbee Survey [22], in the 2010 and 2011 academic year, the female and minority population in the Computing Sciences was around 11% in the USA which is well behind the population ratios between the years of 2000 and 2006. The research conducted by Garner [9] attribute students frustration with Computing to the difficulty experienced while programming and writing code; mathematical maturity, and the substantial investment of time spent in front of a computer. Also, the strong belief that Computing is a competitive and "hard" field is an additional discouraging reason for female and minority students which push them away from embarking on a computing major.

In an effort to attract more students and increase retention rates, several different strategies have been used [20, 5]. Several Computer Science departments offer scholarships [5], companies resort to providing paid internship opportunities and summer training sessions [5], and faculty have attempted to promote their programs and courses using active teaching styles and exposing their students to diverse experiences using exciting research projects [14, 1].

Some faculty target introductory level computer programming courses [2], others aim to target high-school students or seniors. The main rationale behind targeting introductory courses is that some students are particularly prone to change their majors after taking their first introductory computer science courses [15].

We have designed our tutoring software by applying the fundamental tools and principles of learning, namely interactions and exchange of ideas. The field of academics and pedagogy is never an exception for the constantly changing technological environment. With the increase in computerized platforms and virtual replications [10] as common teaching practices, the total number of enrolled students and awarded degrees in the Computer Science has increased in the last couple of years [19]. However, the number is less than the expected students to be enrolled in the sector, according to CRA Taulbee Survey. It also postulates that the number of female and minority population are way behind the figures between the years of 2000 and 2006 [22].

Research studies have revealed that the steady decrease in the number of students in the field of computer science is due to the complication in the understanding of the elementary theories at the apprentice level of programming. In several programming practice environments, there is a shortage of teaching faculty. When students have no idea to solve a problem, they reach a virtual standoff. In many cases, students even fail to consult their fellow class mates as they are hesitant [16]. Silessi et.al [19], mentioned that annoyance while programming, the association of math, and spending most of the time in front of a computer are some of the reasons for the decrease in interest for students in the computer science area.

Providing a programming environment in the tutoring system will aid the students to practice the learned concepts easily and facilitates them to accustom to the environment. The Interactive Bash Shell Tutoring System (iBaTs) is an "adaptive tutor" that provides the fully functioning shell using a console, thus reducing the dependency on the graphical interface gradually. It allows the user to select the topic of their choice. It also provides the flexibility of depicting the progress of contents and the test module aids to characterize the concepts learned by the student.

In section 2, we discuss the work done previously in the literature to address these problems and the tools created to assist the beginner programmers. In section 3, after estimating the benefits and shortcomings of different tools, we discuss the development of our new tutoring system. In section 4, we outline the test cases and results and finally, conclude the paper.

2. PRIOR AND RELATED WORK

The creation of new technologies has been renovating the human lifestyle at stunning speed, where education is never an exemption. Many researchers are motivated in search of new technological tools that reforms the way of learning. Students, in the present era, are not considering the field of engineering and technology as creative and interesting due to the restricted way of learning [4]. The best way of learning is by putting the acquired knowledge into practice. Despite the different learning programs, offered scholarships and summer training courses [5], the students are inclined to conversion of their majors from Computer Science and related degree plans [15]. To incent students in the field of computing, the preliminary and foundational courses should be made more interesting to students [19], which support to retain the talent and creativity for new and innovative ideas.

Marasco and Behjat [13] have proposed a multidisciplinary framework for teaching electronics which include intensive use of WebLabs [13]. In chorus, distance learning, the concept of encouraging students to study by themselves, the application of practical activities have originated [4]. On the other hand, the laborious work while programming is hindering the curiosity of students in programming environment consequently reducing the number of skilled employees in the IT sector [20]. The new method of education with improved classroom and acoustic environments, the use of smart computers, the proposition of real-life challenges encourages students to learn in an appealing fashion. As a result, several novel collaborative tools have been implemented, which aid to increase the interest of students and focus on the concepts instead of memorizing syntax.

Del Fatto et al. [8] have used extreme apprenticeship to measure student performance for learning Bash programming. Extreme apprenticeship was first introduced by Vihavainen [21] in introductory level programming courses, that is, on learning a task as apprentices, by observing how an experienced person performs it. In this work, the experienced person supports the students using the Moodle Learning Management System in an asynchronous manner. Authors found out that with this methodology students performed better in many programming exercises compared to the traditional style of education [8].

BashDungeon is an educational game for teaching UNIX commands. The game screen consists of three parts, the dungeon visual representation, and simulated command line prompt representing the bash learning environment, and menu items. Once the user writes the commands using the keyboard, the game translates the commands into character actions, according to the playing metaphor. To make progress in the game, the user must answer and handle several questions and challenges. These challenges must be solved using bash commands [6].

Dann and Cooper [7] have noted that a 3D programming tool built using the Python language, named *Alice*, and has helped students to transit into the programming environment. It consists of a GUI (Graphical User Interface), which controls and provides a pictorial representation of processes [7]. *Bricklayer* is another approach for high school students; the tool is developed in Javascript and operates on a web browser but generates source in the C language [19].

To develop graphical applications in 2D, a highly interactive tool is designed by Poul Henriksen and Michael Kolling [11], a member of the *BlueJ* team, known as *Greenfoot.* It is an extension of the *BlueJ* tool that eases the overall time needed to learn the principles of design of applications [11]. *Jpie* is a tool, which uses many abstracts and constructs directly. It also facilitates the users with drag-and-drop functionality [2]. It provides visual construction of Java and supports live development of running programs. With immediate feedback flexibility, the tool aids the students to understand the theories without losing hands-on skills [17].

Flow models are other advantageous visual programs that envision the algorithm of any given problem. Over 95% of students selected flow model representation over the algorithmic representation, as it reduced syntactic complexity allowing the students to focus on solving the problem. *RAPTOR* is one of the most used flow model tool which incorporated 3D graphics of *Alice* with the conception of control flow [17]. The Air Force Academy had replaced Ada95 and MATLAB with RAPTOR in its CS0 course and found that the students have reported better performance [3].

Karel is a robot that is considered a CS0 tool. Stanford University has adopted the technology of using Jkarel installed under the Eclipse environment [12]. *Jeero,* developed by Sanders, is a narrative tool, similar to Karel, prominent in control structures, methods, and objects [12]. *Lego MindStorms* is an online robotic tool that helps students to build a robot in different programming languages like C++, Java, etc. With this approach, the usual CS lab experience is converted into an active learning session [18].

Dragon Drop Pictorial Programming is one more tool, designed for the introduction-level programming students, which improved the ability of students to write a program [19]. The tool appeared to be more appealing to the students. Silessi et al. [19], mentioned that in the surveys, they found that over 50% of the students have improved their performance.

In addition to the above tools, there are some virtual environments such as Peer Instruction and Simulation Tools, Avatar-based proposals for teaching students. The use of a virtual environment within computer-aided learning has become the prime research agenda [4]. The practice of flipped classrooms has become frequent in recent years, especially due to its proven benefits [4].

All the tools mentioned above are pertaining to the improvement of programming skills for novice programmers. The main reason for developing such tools is to encourage students to continue in the field of Computer Science. Each tool specifically consists of its own pros and cons, but these programs share most of their benefits and strengths. The tools like *Greenfoot* and *Dragon Drop Pictorial Programming* have proven that the students grasp of fundamental concepts, programming logic and the level of confidence has increased [19]. The most common and prominent advantage shared by all these above mentioned tools is that they have a friendly user interface that supports the user to program and interact with the system with ease.

Despite these numerous enhancements in the programming tools for the Windows environment, there has been relatively little that has been offered for the education and training of the various shells under the UNIX environment. Although there are different bash shell tutorials offered by a number of people and companies on the Internet, the tools we discussed earlier under the Windows environment would ease the user and help the student improve her programming skills. There are no such comparable tools in the UNIX environment. It is this deficiency that we have sought to fill with our tool.

3. METHODOLOGY

The goal of the Interactive Bash Shell Tutoring System (iBaTs) is to create a fully functioning shell as a graphical user interface (GUI). Primarily, the tutor aims to eliminate the discomfort of novice programmers but ultimately helps the user to get acquainted with the console application.

3.1 Features of iBaTs

The primary features of the proposed methodology are as explained below:

- To increase the retention rate of women and underrepresented groups in computing and Cyber Security.
- *Multi-panel web-based system:* The first panel displays the audio/video of the current chapter, the second panel is an interactive fully functioning bash shell, and the third panel shows the available chapters and progress of the user.
- Increase the use of command-line tools for information assurance, incident response security.
- Equip beginning students to interact with their computing device as a real programmer using power tools instead of being dependent on a graphical interface.
- Understands the basic concepts of directories, file systems, redirection, pipes, file I/O essential to the foundations of system and cyber security.
- The tutor can be used as supplementary software in most of courses related to computing, security, information assurance etc.

- Will be made available as a stand-alone installation tool featuring regular updates of course materials and tutorials.
- Interactive quizzes/exams featuring feedback, providing correct answers and progress of students for the content.
- Our tutor is *extensible* - new updates and material/ content can be added to the tool such that the materials are not outdated or irrelevant.
- Finally, the iBaTs system is adaptive. The quizzes and materials are designed to reinforce those concepts that a student finds the difficulty with and can be moved quickly in the areas where student acquired the mastery.

3.2 Implementation
The major task of the tutoring system is to transform the bash shell into an interactive stand-alone platform. Figure 1 illustrates the implementation of iBaTs. The adaptive system helps the user to choose the chapter to open a particular tutorial. It facilitates the user to select the desired mode of tutorial, such as A/V (Audio/Video), presentation, text, etc. Based on the selection, the tutorial is displayed on the screen. The tutoring system is initially composed of chapters containing basic concepts of interacting with a computing system, such as file systems, directories, permissions, etc. Every chapter is composed of topics offering video tutorials, PowerPoint presentations, Word documents, text and PDF files. With the completion of each topic the student then proceeds to complete the test to emphasize their understanding.

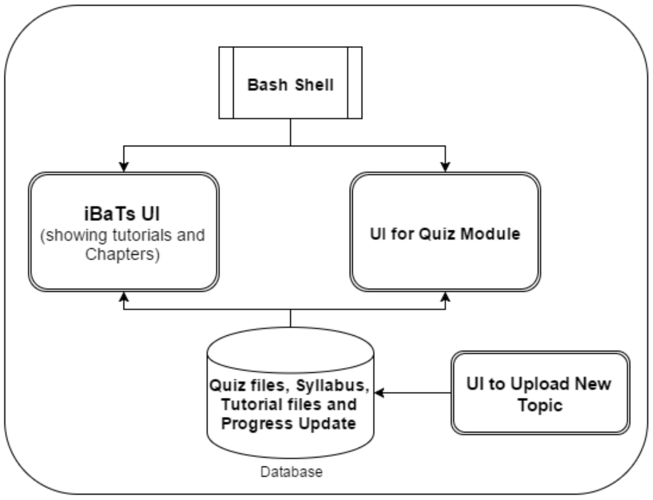

Bash Shell
iBaTs UI
(showing tutorials and Chapters)
UI for Quiz Module
Quiz files, Syllabus, Tutorial files and Progress Update
Database
UI to Upload New Topic

FIGURE 1: Architecture of iBaTs.

An in-built Bash Terminal is available in the tutorial that helps the user to practice and execute the commands learned from the tutorial. The tool also provides the flexibility to upload new topics into the learning system, thus allowing the tool to be updated with new, updated and relevant materials from the instructor, publisher or other sources (in the event that this tool is adopted by textbook publishers and they produce content for the tool, integrated with a learning management system such as Blackboard).

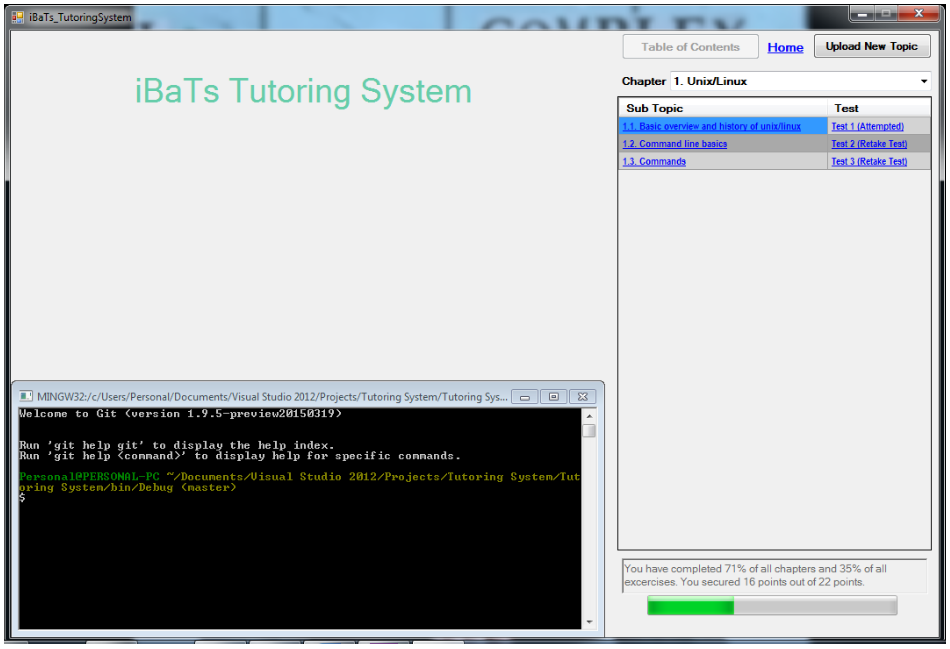
3.3 User Interface of iBaTs
The diagram shown in Figure 2 illustrates the user interface of the iBaTs tutoring system. The aim of the tool is to minimize the frustration experienced by beginner programmers and shell users. The tool first instantiates the Bash shell.

3.4 Upload New Topic
The Upload New topic button allows the user to upload new topics into the tool. Figure 3 shows the user interface for uploading a new topic. It is required that the chapter number and the topic

number should be the same to upload the topics. The tool does not require all the files to be uploaded but at least one file should be given to register the chapter into the database. Whenever the new topic is uploaded it is directly populated in the chapter list on the main UI of iBaTs. The *Home* link allows the user to navigate from Upload new topic form to the main page.

3.5 Test Module

Each and every topic is provided with a test module to reinforce the understanding of the student. Figure 4 represents the user interface of the test module. The test form also consists of the bash shell to aid students while performing exercises. A threshold value is maintained to assure that the user obtains the minimum score and notifies the user to retake the test if required.



Chapter 1. Unix/Linux

Sub Topic	Test
1.1. Basic overview and history of unix/linux	Test 1 (Attempted)
1.2. Command line basics	Test 2 (Retake Test)
1.3. Commands	Test 3 (Retake Test)

You have completed 71% of all chapters and 35% of all excercises. You secured 16 points out of 22 points.

```
MINGW32:/c/Users/Personal/Documents/Visual Studio 2012/Projects/Tutoring System/Tutoring Sys...
Welcome to Git (version 1.9.5-preview20150319)


Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

Personal@PERSONAL-PC ~/Documents/Visual Studio 2012/Projects/Tutoring System/Tut
oring System/bin/Debug (master)
$
```

FIGURE 2: Graphical User Interface of iBaTs.

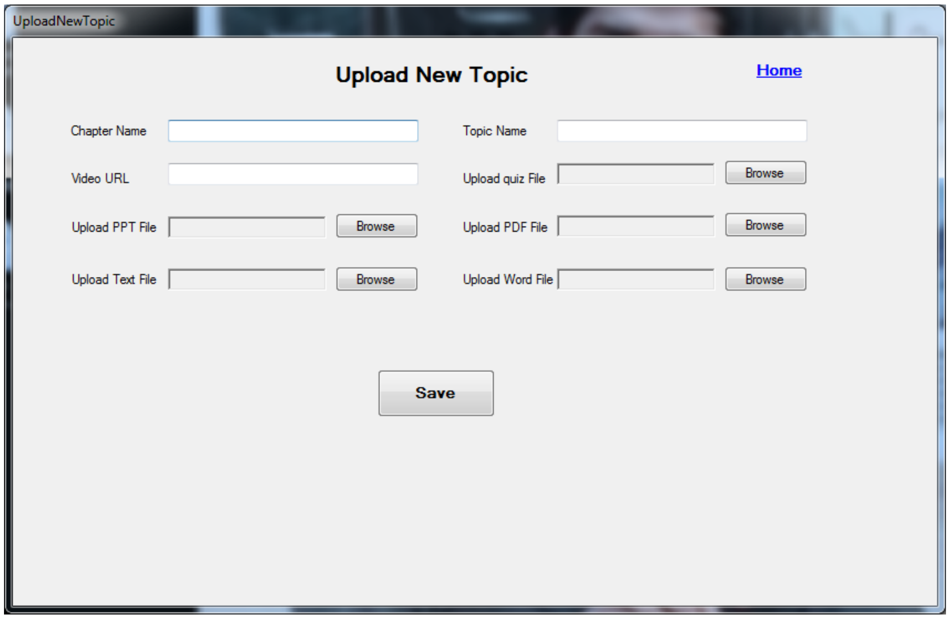
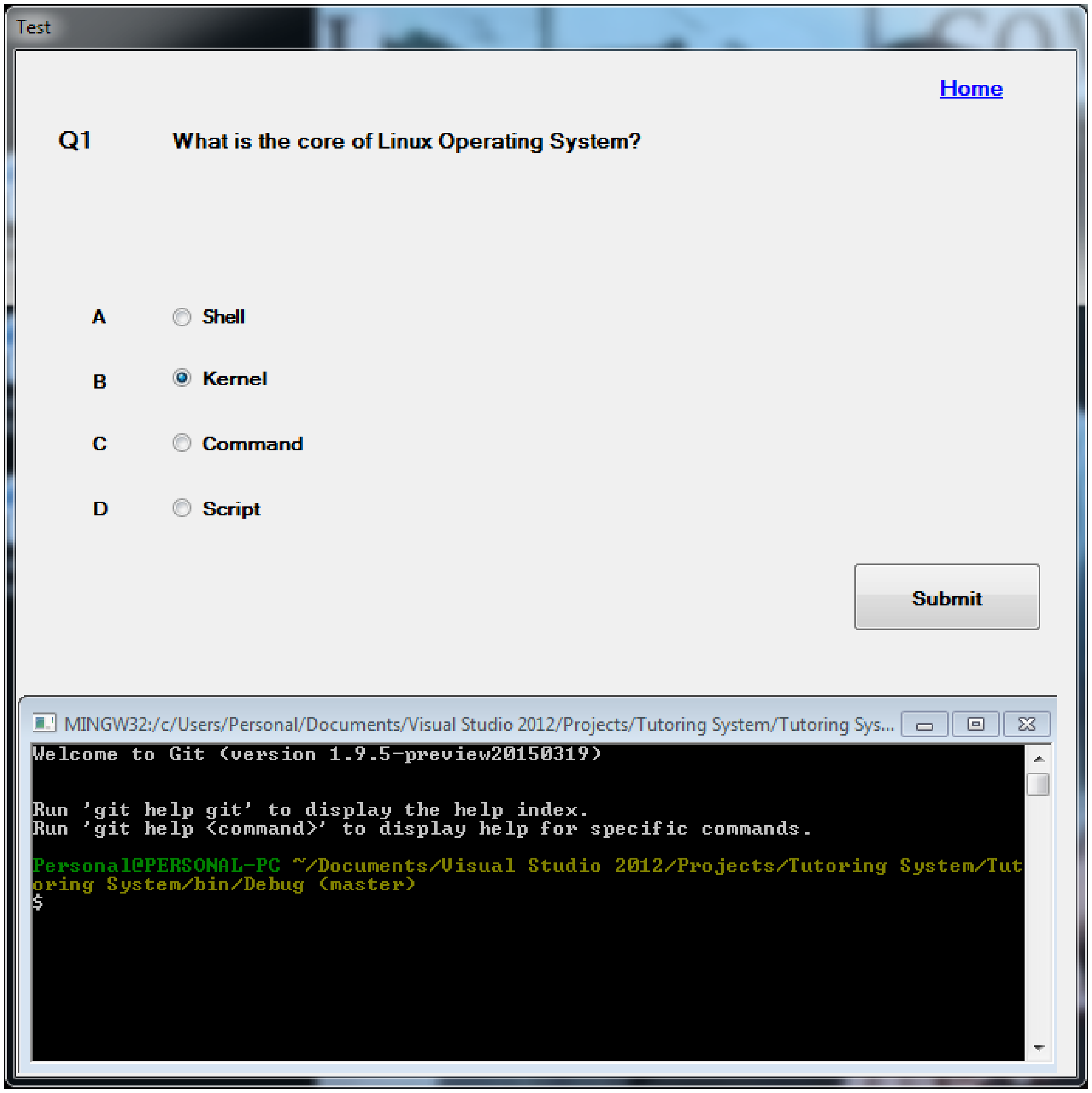


FIGURE 3: New Topic Upload Interface.



Q1 What is the core of Linux Operating System?

A ( ) Shell
B [✓ Kernel]
C ( ) Command
D ( ) Script

```
MINGW32:/c/Users/Personal/Documents/Visual Studio 2012/Projects/Tutoring System/Tutoring Sys...
Welcome to Git (version 1.9.5-preview20150319)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

Personal@PERSONAL-PC ~/Documents/Visual Studio 2012/Projects/Tutoring System/Tut
oring System/bin/Debug (master)
$
```

FIGURE 4: Graphical User Interface for the Test Module.

4. SOFTWARE REQUIREMENTS

The tool requires the students' host system to have the Microsoft .NET framework version 4.0 or higher, with Microsoft Office 2010 or higher version and Adobe Flash Player, version 18.0 or higher.

5. CONCLUSION

The proposed tool alleviates the struggle faced by the beginner student who attempts to master the command-line/console/Bash Shell. With many stand-alone tools available, iBaTs is the one unique tool that provides a shell interface in the Windows environment, to the best of our knowledge. It also helps the users to transit from the graphical user interface to use a console/shell application seamlessly and help the student accomplish several basic computing and elementary security tasks on the shell. The tool can be used as supporting software when adopted by various instructors due to its extensibility and compatibility. Our tool attracts the digital/cyber forensics students along with minority and female students into the field of computing, system, and network administration as well.

6. ACKNOWLEDGEMENTS

The authors would like to thank the support offered by the Office of Research and Sponsored Programs at Sam Houston State University via the grant ERG456914. The authors are solely responsible for the views expressed in this paper, which do not necessarily reflect the position of the supporting organization.

7. REFERENCES

[1] Betancur, J. A., Rodriguez, C., and Ezparragoza, I. (2011). An undergraduate collaborative design experience among institutions in the Americas. In *Proc. 8th WSEAS International Conference on Engineering Education (EDUCATION'11)*, pages 263-265.

[2] Blaho, M.-F., Fodrek, M., and Murgas, P. (2012). J.: Students perspective on improving programming courses. *International Journal of Education and Information Technologies.-ISSN*, pages 2074-1316.

[3] Carlisle, M. C., Wilson, T. A., Humphries, J. W., and Hadfield, S. M. (2004). Raptor: introducing programming to non-majors with flowcharts. *Journal of Computing Sciences in Colleges*, 19(4):52-60.

[4] Cinto, T., Leite, H. M., Peixoto, C. S., and Arantes, D. S. (2014). Virtual learning environments: Proposals for authoring and visualization of educational content. *International Journal of Digital Information and Wireless Communications (IJDIWC)*, 4(3):387-400.

[5] Cohoon, J. M. (2001). Toward improving female retention in the computer science major. *Communications of the ACM*, 44(5):108-114.

[6] Corda, F., Onnis, M., Pes, M., Spano, L. D., and Scateni, R. (2019). Bashdungeon. *Multimedia Tools and Applications*, 78(10):13731-13746.

[7] Dann, W. and Cooper, S. (2009). Education Alice 3: concrete to abstract. *Communications of the ACM*, 52(8):27-29.

[8] Del Fatto, V., Dodero, G., and Gennari, R. (2016). How measuring student performances allows for measuring blended extreme apprenticeship for learning bash programming. *Computers in Human Behavior*, 55:1231-1240.

[9] Garner, S. (2005). The cloze procedure and the learning of programming. In *International Conference on Learning, Granada, Spain*.

[10] Hanzu-Pazara, R. and Barsan, E. (2010). Teaching techniques-modern bridges between lecturers and students. In *7th WSEAS International Conference on Engineering Education, Corfu Island, Greece, published in Latest Trends on Engineering Education*, pages 176-181.

[11] Henriksen, P. and Kolling, M. (2004). Greenfoot: combining object visualization with interaction. In *Companion to the 19th annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications*, pages 73-82. ACM.

[12] Klassen, M. (2006). Visual approach for teaching programming concepts. In *Proceedings of the 9th International Conference on Engineering Education (ICEE 2006)*, pages 23-28.

[13] Marasco, E. and Behjat, L.(2013). Integrating creativity into elementary electrical engineering education using cdio and project-based learning. In *Microelectronic Systems Education (MSE), 2013 IEEE International Conference on*, pages 44-47. IEEE.

[14] Marin-Garcia, J. A. and Mauri, J. L. (2007). Teamwork with university engineering students. Group process assessment tool. *Feedback*, 4:10.

[15] McDowell, C., Werner, L., Bullock, H. E., and Fernald, J. (2006). Pair programming improves student retention, confidence, and program quality. *Communications of the ACM*, 49(8):90-95.

[16] Phuong, D. T. D. and Shimakawa, H. (2008). Collaborative learning environment to improve novice programmer with convincing opinions. *WSEAS Transactions on Advances in Engineering Education*, 5(9):635-644.

[17] Powers, K., Gross, P., Cooper, S., McNally, M., Goldman, K. J., Proulx, V., and Carlisle, M. (2006). Tools for teaching introductory programming: what works? In *ACM SIGCSE Bulletin*, volume 38, pages 560-561. ACM.

[18] Ricca, B., Lulis, E., and Bade, D. (2006). Lego mindstorms and the growth of critical thinking. In *Intelligent tutoring systems workshop on teaching with robots, agents, and NLP*. Citeseer.

[19] Silessi, S., Varol, H., and Varol, C. (2013). Non-computer science majored women students perspective on a pictorial programming environment. *International Journal of Education and Information Technologies*, 7(2).

[20] Styron Jr, R. (2010). Student satisfaction and persistence: Factors vital to student retention. *Research in Higher Education Journal*, 6:1.

[21] Vihavainen, A., Paksula, M., Luukkainen, M., and Kurhila, J. (2011). Extreme apprenticeship method: key practices and upward scalability. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education*, pages 273{277. ACM.

[22] Zweben, S. (2011). Computing degree and enrollment trends. *Computing Research Association*.