WIDLE: Web-based Interactive Development and Learning Environment

Peter Bui, Charles Volzka, Lucas Novoa
Computer Science
{buipj,volzkacj,novoalg}@uwec.edu

1 Project Proposal

One of the biggest obstacles for beginners learning how to program is their lack of familiarity with the systems commonly used in programming. While many students are familiar with operating systems such as Windows or Mac OS X, few have ever directly used a Unix-like system such as Linux, which requires utilizing an command line interface similar to that found on old DOS systems. Even routine tasks such as managing, viewing, and editing files requires learning a number of commands and different programs without the comfort of graphical interface. Because of this lack of familiarity and the steep learning curve, many students often struggle with mundane file operations rather than focusing on higher level concepts and ideas.

However, because of the power and flexibility of Unix systems, command-line oriented systems such as Linux are quite common and thus are used widely both in industry and academia (e.g. despite their graphical appearance, iOS and Android are Unix-based and developers typically use command line tools to program them). Therefore, it is imperative that students use and become familiar with such Unix systems.

The goal of this project is to build a web-based interactive development and learning environment (WIDLE) that allows users to interact with a remote Linux system using a standard web browser. This system will enable users to perform the following operations on a remote Linux system via their web browser:

1. **File Management**: Copy, rename, and move files graphically.
2. **File Transfer**: Send and receive files between the remote machine and the local workstation.
3. **File Viewing**: View files such as audio, images, and video directly within the browser.
4. **File Editing**: Create, edit, and save text files directly in the browser.
5. **Command Shell**: Execute and view the results of programs running on the remove Linux system.

Further explanation of the significance of these features is provided in the next section. The hope of this project is that once WIDLE is available, students will be able to utilize this tool to take advantage of Linux

systems without having to immediately dive into the command line. That is, this would serve as a transition tool that allows students to leverage their existing familiarity with graphical interfaces and web browsers in the context of programming Unix-like systems.

2 Methodology

Although it is possible to have local graphical Unix systems, many organizations such as UW-Eau Claire choose to have local Windows workstations and remote Linux servers. To access these remote Unix systems, users run a local terminal application such as PuTTY[1] on their Windows machine and then connect over the network to the remote Linux server using the SSH protocol [2]. Because the user is using a terminal, they are restricted to executing textual command line programs (i.e. no graphics and no mouse support).

While this is not a problem for experienced and advanced users, it is quite a culture shock for beginners, given their limited familiarity and toolset. For instance, managing files requires memorizing the syntax of commands such as `cp`, `rm`, `mv`, and `ls`. Transferring files between the local machine and the remote system requires a separate application that is not standard on Windows machines. Viewing non-textual files (i.e. multimedia such as audio, images, and video) requires downloading the file first and then viewing it with a separate local Windows application. Editing a file requires learning a text editor such as `vim` or `emacs` or `nano`, which do not support typical keybindings and provide minimal support for the mouse. Having to learn all of these things while also tackling high-level Computer Science concepts is difficult and often frustrating for beginners. Even for advanced users, the requirement of downloading files before viewing them can be quite restrictive and burdensome for rapid development and debugging.

Our approach is to abstract these operations behind a web interface. We will develop an RESTFul [4] application in Python [3] that users can execute to provide a remote access web portal to their files on a remote Linux system. With this software, wsers will be able to take advantage of the web browsers rich multimedia and graphical capabilities to send remote commands to the web portal which will receive these commands and perform the operations on the remote system on behalf of the user. For security, we will automatically generate a password the user must enter in order to access the portal and we will encrypt all traffic using SSL. Interactivity will be supported by using AJAX-based technologies such as jQuery [6] and other JavaScript libraries.

3 Context and Significance

WIDLE fits in with ongoing trends in both Computer Science education and industry of developing easy-to-use graphical interfaces that allow beginners to harness existing powerful but difficult-to-use systems. For instance, Scratch [5] provides a simplified graphical interface for programming games and mobile applications. In the high performance computing realm, there are a variety of projects focused on providing web portals [7, 8] to specific scientific applications and platforms. Likewise, there are now a growing number of cloud-based services for remote access to Linux systems such as Nitrous.IO [9].

Our project is different in that it is designed as a general purpose interface to a Linux system rather than a portal to a specific tool or collection of programs. Moreover, our application is to be run directly by the user on-demand on a system they own or have access to. That is, rather than provide a service, we will give users the application directly and allow them to manage and execute their own web portals.

4 History

This project has not been funded by ORSP previously. The origin of this project came from observing the struggles of students in a variety of CS classes and determining that there is a need for a tool that allows beginners to easily and effectively harness the power of Unix systems without having to learn all of the command line interface at once. Because of this need, the faculty mentor has written a proof-of-concept prototype that shows that most of the proposed functionality is possible, but has yet used it in a course.

5 Dissemination

We hope to present the results of our research in the following venues:

1. A paper at the Midwest Computing and Instruction Symposium (MICS).
2. A poster or presentation at the Celebration of Excellence in Research and Creative Activity (CERCA).

If appropriate, we will also submit a poster or paper to other Computer Science workshops or conferences. Additionally, we plan on open sourcing our software so the other members of the Computer Science education community can utilize our tools. Moreover, we plan on using this platform in future offerings of CS 163 Introduction to Programming in C/C++, CS 170 Computing for the Sciences and Mathematics, and CS 252 Systems Programming.

References

[1] PuTTY. http://www.chiark.greenend.org.uk/~sgtatham/putty/.

[2] OpenSSH. http://www.openssh.com/.

[3] Python. https://www.python.org/.

[4] C Pautasso, O Zimmermann, F Leymann. "Restful web services vs. big'web services: making the right architectural decision". Proceedings of the 17th international conference on World Wide Web, 805-814.

[5] Resnick, Mitchel, et al. "Scratch: programming for all." Communications of the ACM 52.11 (2009): 60-67.

[6] jQuery. http://jquery.com/.

[7] Klimeck, Gerhard, et al. "nanohub. org: Advancing education and research in nanotechnology." Computing in Science and Engineering 10.5 (2008): 17-23.

[8] Bui, Hoang, et al. "Experience with BXGrid: a data repository and computing grid for biometrics research." Cluster Computing 12.4 (2009): 373-386.

[9] Nitrous.IO. http://www.nitrous.io.