University of Zurich[UZH]

Department of Informatics

Binzmühlestrasse 14
CH-8050 Zürich-Oerlikon
Switzerland

Prof. Dr. Harald C. Gall
Software Evolution and
Architecture Lab

Phone +41 44 635 43 35
Fax +41 44 635 68 09
gall@ifi.uzh.ch
http://seal.ifi.uzh.ch

Herr
Qasim Warraich
Längassstrasse 54
3012 Bern

Matrikel-Nr. 18-787-796
qasim.warraich@uzh.ch

March 13, 2022

Master's Thesis Specification

"CLI Tutor"

Introduction

Despite the arguably dated appearance, difficult learning curve and practical non-existence in the general personal computing space, Command Line Interfaces (CLIs) have more than stood the test of time in the software development world. There are a multitude of extremely popular tools and applications that primarily focus on the command line as an interaction medium. Some examples include version control software like 'git', compilers and interpreters for programming languages, package managers and various core utilities that are popular in areas such as software development, scripting and system administration.

As mentioned before, the use of the command line as an interaction paradigm has effectively disappeared from a mainstream personal computer usage perspective. This contributes greatly to the intimidation factor and learning difficulty for those interested in getting into software engineering or system administration. This unfamiliarity, paired with the inevitability of usage of CLIs in the development space highlights a need to make the command line more accessible to new users for whom text-based interaction with their computer is an alien concept. In recent years interactive learning utilising tools such as sandboxed environments have been gaining in popularity and have the potential to be a suitable medium for learning command line basics through actual usage, examples and practice.

The goals of this Master's thesis

This thesis aims to determine whether an interactive learning method may ease the introduction into command line interfaces for novice users, particularly mitigating the 'scare factor' experienced by first-time users. We do so by creating a forgiving CLI with the goal of teaching topics such as shell scripting basics and Unix-like core utility usage through the use of interactive examples. We draw inspiration from the 'vimtutor' [2] utility shipping alongside the popular terminal based text editor *Vim*.

The proposed tool shall allow for opt-in analytics that are sent back to a data collection service for the purpose of learning which mistakes are most commonly made, and to improve the tool accordingly. To validate the tool and answer our research questions, a user study will be conducted, most likely with bachelor students at the University of Zurich. A secondary goal is to embed the learning tool into a prototypical web application in order to make it more accessible and portable.

Research questions

We preliminarily define the following research questions:

- Are there identifiable patterns of difficulty when it comes to adopting CLIs? Can the 'indimidation factor' be pinned down?
- How should an interactive learning tool be designed to mitigate the difficulty and indimidation factor of learning CLIs?
- How can a 'forgiving' shell be implemented on top of an existing shell to enable the transition from learning to real-world usage?
- Is the interactive tool more effective than text based learning methods?
- Are novice CLI users more likely to continue using CLI interfaces after using such a tool?

Tasks

Literature review. A look into some of the existing work performed in this space, in order to ascertain what the core difficulties and issues that would need to be tackled are.

Outlining the curriculum. Deciding what lessons and examples the tutor utility should comprise. This will be done by looking at existing learning tools, guides and through insights gained from the literature research.

Development of the forgiving shell. The task here is to implement a CLI or TUI interface that allows for mistakes and can gently guide the user toward a correct answer rather than producing intimidating or cryptic errors as using a traditional shell directly might do. The 'GoCui' library might be an appropriate candidate for building a tool such as this.

Development of the analytics service. The CLI tutor shall include an opt-in feature that allows users to share their input and output logs with an analytics service that allows us to investigate common mistakes and usage patterns to make improvements to the tool.

Web tool Embed the teaching tool into a website for demonstration purposes and to gather more data from a broader audience.

Validation Performing a user study in order to measure the effectiveness of such a tool.

Milestones

Deadline	What (*thesis-related*)
March 14th	Official start of thesis.
March 20th	Literature review complete. *Related work section written. Research questions defined.*
March 24th	First draft of curriculum completed.
March 29th	Curriculum defined. *Introduction and curriculum written.*
March 29th	Development of tool begins.
April 20th	Interaction framework completed.
May 10th	Majority of lessons implemented.
May 25th	Analytics service implemented. *Method section mostly written.*
June 5th	Web demo implemented.
June 15th	Validation and study defined and begun.
July 10th	Data from web users and study participants collected. *Method section completed*.
July 20th	Analysis of findings completed. *Results and future work sections mostly completed*.
August 14th	Tool, documentation *and thesis* finalized.
September 14th	*Final thesis due date*.
September	Presentation including demo.

General thesis guidelines

The typical rules of academic work must be followed. In [1], Bernstein describes a number of guidelines which must be followed. At the end of the thesis, a final report has to be written. The report should clearly be organized, follow the usual academic report structure, and has to be written in English using our s.e.a.l. LaTeX-template.

Since implementing software is also part of this thesis, state-of-the-art design, coding, and documentation standards for the software have to be obeyed.

Effective feedback can only be provided to the student if the thesis draft is handed in well before the final deadline!

The diploma thesis has to be concluded with a final presentation for the members of the Software Evolution and Architecture Lab (s.e.a.l.).

Special remarks

Copyright. In accordance with current regulations, the student retains the copyright to his work, while providing a non-exclusive, non-revocable, time-unlimited license for it to the university. For this particular thesis, the student intends to keep all source code public so as to provide maximum accessibility to the proposed learning aid.

Responsible assistant: Dr. Carol V. Alexandru-Funakoshi

Signatures:

Student Name Qasim Warraich

References

[1] A. Bernstein. So what is a (diploma) thesis? a few thoughts for first-timers. Technical report, Dynamic and Distribution Information System Group, University of Zurich, 2005.

[2] M. C. Pierce, R. K. Ware, C. Smith, and B. Moolenaar. vimtutor - the vim tutor, Nov 2019.