The Combination of Skills Training for IT Administrators and Programmers[⋆]

Šárka Vavrečková

Silesian University in Opava, Bezručovo nám. 13, Opava, Czech Republic
sarka.vavreckova@fpf.slu.cz

Abstract. Each ICT graduate should be quite familiar with different areas of computer science – programming, administration of operating systems, information systems, databases, network administration, etc., because in many ICT professions, several different areas of computer science intersect. This trend of blending different ICT areas in the professions should be reflected in education as well.
A teacher who teaches the same group of students in several subjects has the opportunity to interconnect these subjects. Probably the widest options are offered by combining a programming subject with any other, but also in other pairs of subjects some relations can be found.
In the paper we discuss tools and methods for the practical teaching of some subjects and the possibilities of their interconnection. Command line skills training tools can be used to teach operating systems, while in programming subjects, students may be interested in how the similar tools are created. Programming command-line training tools requires compiler programming skills, and students working on a simplified command-line interpreter will practice the command syntax for some selected operating system. It is also beneficial for students to search for these tools, not only for the searching itself, but also for gaining orientation in license conditions and correct understanding of concepts such as open-source vs. closed-source software, license, availability, freeware vs. free software, etc.

Keywords: Teaching · Training tool · Programming · Operating system · Compiler.

1 Introduction

The current demands on ICT graduates are based not only on knowledge and skills in ICT, but graduates are also required to be maximally independent and self-sufficient at work, searching for the necessary information and self-study. The field of ICT is growing very fast and if graduates do not meet these requirements, they are not able to keep up with development in the field. Another requirement is the ability to collaborate in teams. If students only sit in a classroom and

[⋆] This work is funded by the project to support foreign mobility of academic staff and strengthen international bindings at the Silesian University in Opava.

passively consume teacher's lecture, they have less chance of finding a good job after studies and being successful at work. These issues are discussed in [1].

There are many different approaches to the practical teaching of operating systems. Some of them are described, for example, in [5,9,14]. It is often a discussion of which operating system to use for teaching (open-source Linux or some lightweight system designed for teaching purposes such as MINIX, XINU, Nachos, etc.), usually the concept and objectives of a course are also indicated, and sometimes a bit of statistics. Some papers present methods of teaching operating systems using virtualization (one interesting paper about IVE, Integration of Virtualization in Education, can be found in [8]) or special applications, whether desktop, web, or even mobile (e.g. [3]).

The study of operating systems can be divided into three levels:

1. The first level is orientation in the graphical and text-based environment of common operating systems, their configuration, administration, maintenance, security. This degree is important for all ICT students, especially future administrators and technical support.
2. The second level consists in the study of the architecture of operating systems, synchronization, structures and operations in the kernel of operating systems, memory management, CPU scheduling, management of I/O devices, drivers, security subsystem. This level largely intersects with the previous one and is also related to programmers.
3. The third level is the most challenging: to fully understand all the functions of the kernel, students examine the kernel code and program some parts of the operating system themselves. This level is important for future kernel and driver programmers.

While in the first and second level we deal with existing commonly used operating systems (Windows, Linux, MacOS, Android, etc.), the third level requires the use of lightweight operating systems to make teaching time and didactically manageable. Some of these systems are listed in [5].

In order for students to be able to use a certain command in the right way at the right time, they must not lack the related theoretical knowledge. For example, if students want to start a process with a lowered priority, they need to know the relationship between program and process, what such priority is called, and which command can be used; if they are to modify the behavior of the memory manager in the Windows registry database, they must understand the role of memory manager and be knowledgeable themselves in the registry.

The objectives of courses in the range of the first two levels could be:

– understand the role of an operating system in the management of resources of computing system, the role of the kernel and system calls,
– learn how to find information and configure a given operating system in the both graphical and text-based interface,
– understand the principles of files, users, process, memory, device and network management in operating systems, understand the issues of synchronization and deadlock,

- perform files, users, process, memory, device and network management,
- understand the principles of security in operating systems,
- configure an operating system to be sufficiently protected.

Of course, we can provide students with study materials and all the tools and applications they will need to study right away. However, this does not force students to actively cooperate in the process of increasing their skills. In addition, the ability to search for information and tools is very important for each ICT graduate.

2 Possibilities of Operating Systems Training

In this paper, we will focus on the first level from the list in Introduction, i.e. training configuration, maintenance and security of the following operating systems:

- MS Windows can be found on many computers of regular users and smaller servers, and therefore it is very important for administrators to handle this system,
- Linux installed (in various forms – distributions) on some servers, clusters, supercomputers, it is freely available, and other UNIX and UNIX-like systems (MacOS, Solaris, FreeBSD, QNX, etc.) are handled very similarly.

Tools for training these skills can be found on the Internet, in various qualities. These are either reference manuals with passive explanations of functionality or parameters of commands, or collections of examples with solutions, or, conversely, complex courses and applications with feedback in varying degrees of didactic quality. Some of these tools are freely available, others commercial.

There are also educational videos on the Internet, the quality of which is very diverse and they do not provide any interactivity.

As for software, some of the tools are freely available, but sometimes only if certain conditions are met. For example, there are tools freely available for non-commercial use only (this is the case with many freeware tools distributed under the EULA license), while software available under the GNU GPL license is really free for use. In this way, students can learn how to find out the license conditions and to understand them.

The following list of resources for various reviews, tutorials, and courses could be much longer, but this paper is not intended to provide a comprehensive overview of such resources.

2.1 Existing Tools for MS Windows

Windows administrators must be able to handle common tasks not only in the graphical interface, but also in the Command Prompt and Powershell. They should know and be able to use commands for working with files, users, processes, devices, registry, power management, performance, security, both for information retrieval and for configuration.

A simple reference manual for commands can be found at the web [19]. This resource is suitable for advanced users, less for students. There are also books, such as [17] – more extensive, but still in the form of a mere summary.

Very interesting tutorial about the Windows PowerShell can be found at the web [6] or [16], which are inventive and entertaining, but with no feedback.

There are interesting courses at the web [11], or certainly ComptTIA A+ certifications, but students don't like these courses as they are commercial and time-consuming.

2.2 Existing Tools for Linux

For Linux/UNIX administrators, the ability to use text-based tools is even more important than for Windows administrators, as these operating systems are often installed with no graphical interface, especially on servers and clusters (however, MS Windows Server can also be installed in the "server core" mode with no graphical interface).

Compared to Windows, Linux and other similar operating systems provide much more text-oriented tools, and almost anything can be found out and configured in a text shell. There is also a choice in text shells (similar to the Command Prompt in Windows), the most used shell in Linux is bash.

In the case of Linux, a quite detailed help for commands can be found inside each Linux/UNIX system and on the Internet. The web [4] provides a sequence of tasks with their solution.

At [10] we can find several courses, the basic set of courses is free. Because the knowledge is divided into several courses, students can be offered a degree that is sufficient and corresponds to their future focus. Free course of Linux is available at the web [13], including instructional videos and exercises. Students can take courses by NDG [15] as well, e.g. the course Introduction to Linux is freely available. Free trial courses on Linux, but also other topics (AWS, Azure,...), can be found at [12].

There are even freely available open-source games that can be used to learn Linux, which is very motivating for current students. Few Linux games are described at the web [7].

3 Programming Training Tools by Students

3.1 Requirements

Making compilers requires (not only) knowledge of advanced programming techniques. If students are to program a simplified compiler that simulates the Windows Command line or bash shell for Linux, they need the following skills:

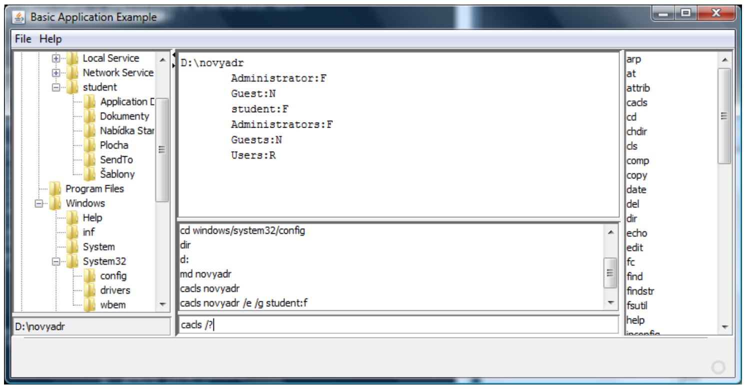
- handling the programming language in which they make the compiler,
- mastering the programming techniques for creating compilers,
- knowing the commands the compiler will process, and being able to find details about these commands, their parameters and common output,

- being able to design clearly arranged interface of the application,
- the application should have a certain didactic quality.

The last requirement would be very problematic if the application should be intended for students who are currently gaining knowledge and experience with the given operating system. But the application is supposed to help and intended primarily for its author (or for the team of cooperating students, authors). Thus, the didactic value is primarily the process of creating the application, all its stages from analysis through the possible planning teamwork, programming of particular parts, until testing.

Before starting to create the own product, a student should look for similar applications or services on the Internet as part of the problem analysis. He or she can be inspired by the discovered tools and at the same time think about what license to use if he or she wanted to distribute the own software product or offer a cloud service.

It does not matter which programming language and development environment students choose. Of course, when working together in a team, they should agree on these parameters.



```
D:\novyadr
        Administrator:F
        Guest:N
        student:F
        Administrators:F
        Guests:N
        Users:R
```

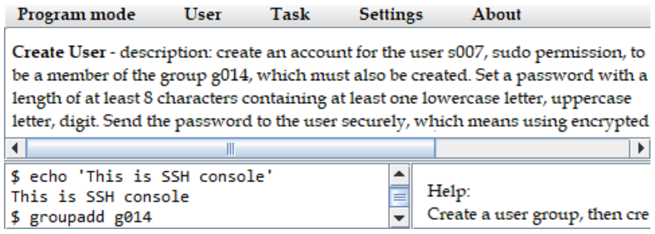
```
cd windows/system32/config
dir
d:
md novyadr
cacls novyadr
cacls novyadr /e /g student:f
```

```
cacls /?
```

Fig. 1. Simple Application for Windows Administration Training

Figure 1 shows a sample Java application for training Windows Command Prompt commands. The left pane contains the structure of folders in which the user moves, the right pane holds a list of supported commands (the user can take help for commands), in the middle there is the output of the entered commands, the pane for previously entered commands (history), and the user enters the commands at the bottom. Some of these panes can be hidden.

Skillful and motivated students can further improve their application, for example by separating the student and administrative access and creating two modes: working and testing. Figure 2 shows a part of the environment of a more advanced application for training Linux administration. A user fulfils a task, the



Create User - description: create an account for the user s007, sudo permission, to be a member of the group g014, which must also be created. Set a password with a length of at least 8 characters containing at least one lowercase letter, uppercase letter, digit. Send the password to the user securely, which means using encrypted

```
$ echo 'This is SSH console'
This is SSH console
$ groupadd g014
```

Help:
Create a user group, then cre

Fig. 2. More Advanced Application for Linux Administration Training

assignment of the task is in the upper pane, the commands are entered in the left pane, help can be found in the right pane.

The author of the mentioned application (Daniel Valenta) decided to continue to expand the possibilities of the application up to the level of the diploma thesis and gradually a project consisting of a server and a client part was created. The server part is made in Python, the client part in Java SE (multiplatform). The running task creates a problem on the server (damages it) and the user has to fix the problem. There are several modes with various levels of help. The server runs virtually and when the task ends, it returns to the state it was in before the task started, its reinstallation is not required.

3.2 The Role of Teacher

Students should already be able to use programming languages and development environments. Therefore, the teacher leads them methodically rather in the areas of compiler programming techniques and the usability of the resulting application.

Programming a complete simulator accepting all common commands of a given operating system is too demanding and time-consuming, so it is necessary to simplify the assignment for students. For example:

– each student has its own subset of commands and parameters to implement,
– students work in teams,
– there is a pre-prepared part of the application, students add new modules for selected commands,
– there is a pre-prepared part of the application, students correct deliberate mistakes in some existing code.

Nowadays, the great topic is secure programming. Any application that receives text or binary input data and further processes it, could become the target of a hacker attack. This is also the case for all applications that receive and process text commands. Unfortunately, when teaching compilers, there is not enough space on this topic, but it is still appropriate to at least draw attention to this problem. An interesting paper on the detection of forged malicious commands is, for example, at [18].

Another topic students should be pointed out is usability, see [2]. The interface of the application in Figure 1 is less usable than the interface of the application in Figure 2, but its layout more suggests the original Command prompt (entering commands at the bottom below the output of previous commands). However, the layout of the second application may not suggest anything, it is mainly about easy understanding the task and quickly finding the pane into which the commands need to be written.

The role of teacher also includes providing feedback (not only the application should provide feedback to its user, the teacher should provide feedback to the programming student as well). While, for example, a test in some LMS presents feedback to a student immediately depending on how a teacher entered comments on each possible answer, in the case of an application programming, providing the correct feedback is more difficult, even in terms of the amount of time spent.

4 Benefits for Students

The main benefit of this method is easier motivation of students. Students prefer to program applications in which they see practical use and which are of good importance for themselves, than applications whose purpose is just to get an evaluation.

Students acquire knowledge and skills belonging to two or even more different subjects, and these skills are practically oriented. Especially for ICT students, it is very important to be knowledgeable about the manuals and quickly find what they actually need.

For (not only) programmers, it is also beneficial to consider the features of different types of software licenses. It's not just about which license to choose for the own products, but every ICT professional uses various software tools and cloud services under licenses whose terms need to be understood.

The next benefit is the practice of stand-alone work on a larger project, or teamwork, depending on the difficulty of the assignment and the amount of time available to students.

If students are sufficiently motivated and the assignment corresponds to their abilities, the resulting applications are very interesting and students can use them as a demonstration of their skills for future employers.

5 Conclusions

This paper presents one of the ways to enrich and interconnect teaching of two or more subjects at the bachelor's degree level of a university. The fields of programming and operating systems are important for most graduates of ICT study programs and it is necessary to look for ways to increase students' skills in these fields in a motivating and effective way.

The effectiveness of the proposed method is not ideal (especially for the time-consuming supervision and feedback), but for many students it could be a very good motivation and a launching pad to their professional activity.

References

1. Aasheim, Ch.L., Li, L., Williams, S.: Knowledge and Skill Requirements for Entry-Level Information Technology Workers: A Comparison of Industry and Academia. *Journal of Information Systems Education*, **20**(3), pp. 349–356 (2009). Available at: https://aisel.aisnet.org/jise/vol20/iss3/10
2. Bevan N., Carter J., Earthy J., Geis T., Harker S.: New ISO Standards for Usability, Usability Reports and Usability Measures. In: Kurosu M. (eds) Human-Computer Interaction. Theory, Design, Development and Practice. HCI 2016. LNCS, vol 9731. Springer, Cham (2016). DOI: https://doi.org/10.1007/978-3-319-39510-4_25
3. Chakraborty, P. et al.: OSAVA: An Android App for Teaching a Course on Operating Systems. Journal on Engineering Education Transformations, **32**(4), pp. 20–30 (2019). ISSN 2349-2473.
4. Command Challenge, https://cmdchallenge.com/. Last accessed 10 Jul 2020
5. Giraddi, S., Kalwad, P., and Kanakareddi, S.: Teaching Operating Systems-Programming Assignments Approach. *Journal of Engineering Education Transformations*, **31**(3), pp. 68–73 (2018).
6. Introduction to the Windows Command Line with PowerShell, https://programminghistorian.org/en/lessons/intro-to-powershell. Last accessed 10 Jul 2020
7. Kenion, S.: 3 command line games for learning Bash the fun way, https://opensource.com/article/19/10/learn-bash-command-line-games. Last accessed 10 Jul 2020
8. Klement, M: Models of integration of virtualization in education: Virtualization technology and possibilities of its use in education. Computers & Education, vol. 105, pp. 31–43, ISSN 0360-1315 (2017). DOI: https://doi.org/https://doi.org/10.1016/j.compedu.2016.11.006.
9. Kolgatin, O., Holubnychyi, D., and Kolgatina, L.: Systematicity of students' independent work in the course of operating systems. In: *SHS Web Conf.*, vol. 75 (2020). DOI: https://doi.org/https://doi.org/10.1051/shsconf/20207503009
10. Learn the Command Line, https://www.codecademy.com/learn/learn-the-command-line. Last accessed 10 Jul 2020
11. Learn Windows Command Line, https://www.udemy.com/course/learn-windows-command-line/. Last accessed 10 Jul 2020
12. Linux Academy, https://linuxacademy.com/. Last accessed 10 Jul 2020
13. Linux Command Line Basics, https://www.udacity.com/course/linux-command-line-basics–ud595. Last accessed 10 Jul 2020
14. Machanick P.: Teaching Operating Systems: Just Enough Abstraction. In: Gruner S. (eds) ICT Education. SACLA 2016. Communications in Computer and Information Science, vol 642. Springer, Cham (2016). DOI: https://doi.org/10.1007/978-3-319-47680-3_10
15. NetDevGroup Courses, https://www.netdevgroup.com/online/courses. Last accessed 10 Jul 2020
16. Powershell Tutorial, http://powershelltutorial.net/. Last accessed 15 Jul 2020
17. Stanek, W.: Windows Command Line Self-Study Training Kit. Stanek & Associates, 2016. ISBN 978-1627164542.
18. Threat Research: Obfuscated Command Line Detection Using Machine Learning, https://www.fireeye.com/blog/threat-research/2018/11/obfuscated-command-line-detection-using-machine-learning.html. Last accessed 13 Jul 2020
19. Windows Commands, https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/windows-commands. Last accessed 10 Jul 2020