Let’s WIN LIN
# Table of contents

<table>
<thead>
<tr>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Chapter I – History of Linux.</strong></td>
<td>5</td>
</tr>
<tr>
<td>GNU/Linux naming controversy.</td>
<td>9</td>
</tr>
<tr>
<td>Open Source and Linux Foundation.</td>
<td>10</td>
</tr>
<tr>
<td>BSD License</td>
<td>16</td>
</tr>
<tr>
<td>Costs</td>
<td>18</td>
</tr>
<tr>
<td><strong>Chapter II – Linux a multiuser system.</strong></td>
<td>21</td>
</tr>
<tr>
<td>TTYs</td>
<td>23</td>
</tr>
<tr>
<td>Group management</td>
<td>26</td>
</tr>
<tr>
<td>Virtual Machines</td>
<td>30</td>
</tr>
<tr>
<td><strong>Chapter III – Installation and applications.</strong></td>
<td>35</td>
</tr>
<tr>
<td>Linux Installation</td>
<td>36</td>
</tr>
<tr>
<td>Linux Distributions</td>
<td>37</td>
</tr>
<tr>
<td>Installation of applications</td>
<td>42</td>
</tr>
<tr>
<td>Window managers and desktop environments</td>
<td>47</td>
</tr>
<tr>
<td>Program equivalents</td>
<td>51</td>
</tr>
</tbody>
</table>
Chapter IV – Connecting to Linux

About Hardware ................................................................. 58
Connecting a device to the PC........................................ 60
Hardware compatibility and recognition .......................... 61
The kernel........................................................................... 64

Chapter V – Practical uses................................................ 71

File extensions ........................................................................ 80

Chapter VI – Facing Linux................................................. 81

First Impressions ................................................................. 83
Getting Used .......................................................................... 85
Starting to Enjoy ................................................................. 86
Opinions. ............................................................................... 89
Review....................................................................................... 90
Chapter I – History of Linux

• The creation of Linux
• Licenses
• Costs
Events leading to the creation of Linux

It all started in 1969 when Ken Thompson and Dennis Ritchie, both from AT&T Bell Laboratories conceived and implemented the Unix operating system. It was first released in 1970, and was widely adopted, copied and modified by academic institutions and businesses, because of its availability and portability. The authors of other systems were strongly influenced by the design of this operating system.
In 1983, Richard Stallman started the GNU project with the goal of creating a free UNIX-like operating system. As part of this work, he wrote the GNU General Public License (GPL). By the early 1990s there was almost enough available software to create a full operating system. Because of the failure of the GNU kernel (Hurd) to attract enough attention from developers, the GNU project was left incomplete.

Another free operating system project, initially released in 1977, was the Berkeley Software Distribution (BSD). In 1986, Maurice J. Bach, of AT&T Bell Labs, published *The Design of the UNIX Operating System*. This definitive description principally covered the System V Release 2 kernel, with some new features from Release 3 and BSD.

These factors and the lack of a widely adopted, free kernel provided the impetus for Torvalds's starting his project. Linus was convinced that if either the GNU or 386BSD kernels were available at the time, he surely would not have written his own.
The creation of Linux

In 1991, in Helsinki, Linus Torvalds began a project that later became the Linux kernel. It was initially a terminal emulator, which Torvalds used to access the large UNIX servers of the university. He wrote the program specifically for the hardware he was using and independent of an operating system because he wanted to use the functions of his new PC with an 80386 processor. Development was done on MINIX using the GNU C compiler, which is still the main choice for compiling Linux today (although the code can be built with other compilers, such as the Intel C Compiler).
GNU/Linux naming controversy

The designation "Linux" was initially used by Torvalds only for the Linux kernel. The kernel was, however, frequently used together with other software, especially that of the GNU project. This quickly became the most popular adoption of GNU software. In June 1994 in GNU's bulletin, Linux was referred to as a "free UNIX clone", and the Debian project began calling its product Debian GNU/Linux. In May 1996, Richard Stallman published the editor Emacs 19.31, in which the type of system was renamed from Linux to Lignux. This spelling was intended to refer specifically to the combination of GNU and Linux, but this was soon abandoned in favor of "GNU/Linux".
The Open Source Development Lab (OSDL) was created in the year 2000, and is an independent nonprofit organization which pursues the goal of optimizing Linux for employment in data centers and in the carrier range. It served as sponsored working premises for Linus Torvalds and also for Andrew Morton (until the middle of 2006 when Morton transferred to Google). Torvalds works full-time on behalf of OSDL, developing the Linux Kernels.

On January 22, 2007, OSDL and the Free Standards Group merged to form The Linux Foundation, narrowing their respective focuses to that of promoting GNU/Linux in competition with Microsoft Windows.
Short chronology

- 1983: Richard Stallman creates the GNU project with the goal of creating a free operating system.
- 1989: Richard Stallman writes the first version of the GNU General Public License.
- 1991: The Linux kernel is publicly announced on 25 August by the 21 year old Finnish student Linus Benedict Torvalds.
- 1999: A group of developers begin work on the graphic environment GNOME, which should become a free replacement for KDE, which depended on the then proprietary Qt toolkit. During the year IBM announces an extensive project for the support of Linux.
A free software license differs from a usual software license by granting recipients of the software rights prohibited by copyright law, rights like modifying or redistributing.

In 1989, Richard Stallman created the GNU General Public License (GPL) for the use of the project. Today it is the most popular free software license.

GPL allows recipients to distribute or modify works, but with the requirement of preserving these rights for the following users. Creators of this practicing called it “copyleft”, because it uses copyright law to offer these rights.
At first GPL was a combination of licenses created and used by GNU for earlier software. Before GPL, every software produced by the project had its own license, specific to that program. Despite this fact, these licenses were very similar because of the same basis on which they were written. Stallman succeeded in creating a single license that would make possible for any program to share the same rights of use.
Terms and conditions

Every user of a software that is under the GPL license has the right to be aware of the terms and conditions of the GPL. Any recipient that doesn't disobey the GPL main license has the rights stipulated in the GPL. Moreover, the recipient is allowed to put any price on the program, if he wants to sell it. This fact makes GPL different from other licenses, which prohibit commercial use.

The GPL doesn't allow a recipient to impose “further restrictions on the rights granted by the GPL”. This is one of the reasons why distribution under a non-disclosure agreement is not allowed. Also, when someone distributes a GPL plus his/her own changes, the requirements for distributing the whole work cannot be any greater than the requirements that are in the GPL (copyleft).
Because GPL work is copyrighted, one has no right to redistribute a program, except under the terms of the GPL license. The only requirement for being able to exercise the rights normally restricted by copyright law is to adhere to the terms of the GPL. If one disobeys the terms of the GPL while distributing copies of a work, he/she can be sued by the original author under copyright law.
BSD License

BSD are a set of very permissive software licenses. The first BSD license that was written was used for the Berkeley Software Distribution, a Unix-like operating system that gave the name of the license too.

The original BSD license was created at the University of California. This first version was revised repeatedly and the resulting licenses are called BSD licenses.

BSD is more permissive than GPL, placing minimal restrictions on the redistribution of a software. This means that one can do almost anything he/she wants with the software, even considering it his/her proprietary. This is in contrast to copyleft licenses.
It was told about this license that you can "do what you want with the code as long as you include this notice, don't use our names to endorse your product, and don't sue us".
Costs

Because Linux systems are based on open source software, they provide you greater flexibility than proprietary platforms. Virtually any element of the desktop can be tailored to your specific needs. Unlike Windows, advanced configuration on Linux does not require a separate software stack that costs you additional licensing and training fees. Many people are 100% sure that Linux is genuinely free. Everybody can borrow a CD with Linux from anybody and install as many copies as they like without paying anything for licensing. Therefore, a comparison between the Linux license and the cost of a Microsoft server operating system license should have as result a $0 for Linux to over $800 for Windows, depending on hardware, usage and the number of clients allowed.
The officials from Microsoft still insist on the idea that Windows is cheaper than Linux. As proof, they brought again some arguments on their side to put their OS in a better view. Furthermore, they adduced an upfront cost of $4,000 for Windows 2000 Advanced Server plus $5,000 for the SQL Server, in contrast to Red Hat Enterprise Linux at $2,500 for one year premium support plus DB2 at $7,500, or Oracle at $15,000.

According to a study which examined the costs of running various operating systems over three years, the downtime and maintenance cost areas represent 75% of TCO over a 3-year period, while software acquisition costs represent 7% of TCO over that same period. The Windows technicians, however, only managed an average of 10 machines each, while Linux admins can generally handle several times that, because the network and systems management capabilities in Linux are extremely robust. Red Hat’s business is based on annual subscriptions for OS support - you pay a subscription for every server, every year. And, if you want 24/7 support, you’ll pay more. Even though, the cost of running Linux is roughly 40% that of Microsoft, Linux costs $75,000 over three years, while a Windows deployment costs $190,000.
At large scale, the organizational benefits to Linux do not only come from a one-time capital cost reduction, but from long-term IT cost control coupled with an increase in IT's ability to contribute to organizational productivity. Most important, the overall robustness and stability of Linux operating systems means critical applications stay up and running - keeping your business focused on the work at hand. Investments in Linux continue to be valuable throughout the fiscal years.

Another great advantage alongside with those presented earlier is the multiuser function of the operating system.
Chapter II

LINUX
A system for more users
Multi-user is a term that defines an operating system or application software that allows concurrent access by multiple users of a computer.

All UNIX-like systems provide multi-user features.
TTYs

One of the major benefits of LINUX-based OS is their innate and intuitive multi-user capability, innate in the sense of requiring no third party software to function.

Firstly, Linux provides the capability of starting multiple terminals, or TTYs, on the same machine, which can thereby be accessed by different users without the need for a graphical interface. There are, by default, 6 TTYs; however, the root user can adjust this number, as he can do with all quantifiers in Linux in general.
Furthermore, through a LAN connection, someone on a different machine could access one of your TTYs, without interfering with processes active on the others.
Secondly, Linux comes with a remarkably easy to use feature for creating groups and assigning members to them. This function requires little more than typing several commands, such as 'gpasswd' to add a username to a given group, or 'adduser' followed by a group name to create an account directly in the aforementioned group. This feature comes in very handy on machines working with a large userbase.
SSH

One easy way to connect to other machines in your LAN with Linux is via SSH, the Secure Shell Daemon.

```
buruiana@Sebastian:~$ ssh talivy@192.168.0.101
Password:
Linux vacas.leonardo.local 3.2.0-3-amd64 #1 SMP Mon Jul 23 02:45:17 UTC 2012 x86_64
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Sep 13 10:06:11 2012 from 192.168.0.102
talivy@vacas:~$ 
```
This is done with the simple issuing of the command ssh followed by the name of the user, the @ symbol, and his IP address. Through ssh, users can modify or exchange (via the 'scp' command) files between their stations, provided sufficient permission is granted.
SSH is safe because one cannot log in to the master user (root) without knowing the specific password, and unprotected user accounts, which can be freely accessed through Secure Shell, don't usually have writing permission to sensible files. In addition, if necessary, the root user can always ban another station via 'iptables', reliably cutting all access to his machine from that particular terminal.
Virtual Machines

Most UNIX-like systems also support the installation of multiple virtual machines. A virtual machine is by definition an isolated software implementation of an OS, which is given limited resources.

Virtual Machines are often referred to as VMs.
There are two different categories of Virtual Machines:

1. System Virtual Machines
2. Process Virtual Machines
1. System Virtual Machines

A system VM provides a complete system platform, which is required to fully execute the OS and support multiple processes. Its main advantages are:

Multiple OS can co-exist without interfering with each other.

The possibility of obtaining a different instruction set architecture (ISA), which gives you the chance of testing and comparing different operating systems at the same time.

Easier maintenance, better application management and disaster recovery.

However, System VMs are less efficient than physical machines not only due to the restricted resource input, but because of the indirect link to the hardware.
2. Process Virtual Machines

A process VM runs as an application inside another OS and is only able to support a single process. It is created at the start of the process and is deleted when the program is closed. It uses a far more advanced programming language than the system VMs. Therefore, they are most often created using an interpreter.
VMs on LINUX-based systems

Using virtual machines is much more practical on LINUX systems. These systems, unlike most other well-known OS, such as Windows, which takes considerably more memory, LINUX systems allow more VMs to be active simultaneously, due to reduced memory and CPU consumption.
Chapter III

- Installing Linux (DualBoot)
- Distributions
- Installation of applications
- Window managers and desktop environments
- Program equivalents Windows - Linux
Linux Installation (DualBoot)

For data exchange between a Linux distribution and Windows it is recommended to create a partition that both systems can read and write. First, the hard drive must be partitioned. Then one should make sure that the PC is configured to boot from CD/DVD (BIOS Setup).

If the boot from DVD is successful you should get a graphical installation guide and you should simply follow the steps recommended by this guide.
Linux Distributions

Distributions (for short Distros) are operating systems which contain a Linux kernel, GNU utilities and libraries and a wide collection of software applications. Due to the fact that most of the Linux kernels and software packages are based on the idea of free and open source software, a wide variety of Linux distributions are now available. There are two main types of Linux distributions:

- the commercially backed distributions (such as Fedora-Red Hat or Ubuntu-Canonical Ltd.)
- the entirely community-driven distributions (such as Gentoo or Debian).
Ubuntu

There are many reasons why Ubuntu is one of the most popular distributions with users: a well web-based infrastructure, free support period, professional approach to the end user. Moreover, Ubuntu is the right choice if you are just starting to work with Linux. On the other hand, one would not pick Ubuntu because of the frequent major changes of this distribution, not to mention the fact that it lacks compatibility with Debian.
Perhaps security is one of the most outstanding feature of Fedora distribution. It is also regarded as the most innovative Linux system. One can have problems with desktop usability (the lack of a clear strategy regarding the desktop), since it is not one of Fedora's main priority.
CentOS

Has gained a reputation of a solid and reliable service distribution, being preferred by those who need stability and long-term support instead of the newest software available. Although it does not live up to the expectations of its users regarding security updates and it lacks the newest versions of Linux, the software packages and the Linux kernel are well-tested and stable.
Debian

Developed by more than 1000 volunteers, it is easy to understand that Debian is rich in software packages compiled for 11 processor architectures. However, there are also some disadvantages when using Debian. For instance, the release cycle is quite slow (one stable release every 1-3 years) and sometimes the newest technologies are not included, Debian being a conservative distribution.
Installation of applications

The installation of programs under UNIX-like systems is different than in Windows in the sense that in Linux, you don't always have to download an installation package from a web site to install a program - very often, they're located in your distribution's repositories. A software repository is a storage location from which software packages may be retrieved and installed on a computer.
Repositories may be solely for particular programs or for an entire operating system. Operators of such repositories typically provide a package management system, tools intended to search for, install and otherwise manipulate software packages from the repositories.
There are types of package management systems according to different distributions of Linux:

- **yum** – RedHat Linux
- **APT** – Debian/Ubuntu developed in 1998 by the apt team
- **aptitude** – the predecessor of apt, developed in 2004 by Daniel Burrows

These are accessible with the use of the console and can be used for a normal installation or upgrade of a single program to the task of updating and upgrading an entire operating system.
Types of package management systems

Because they require the use of a console, these applications may seem inaccessible to newcomers, but there are programs that combine the point-and-click simplicity of the graphical user interface with the power of the `apt-get` command line tool. You can install, remove, configure, or upgrade software packages, browse, sort and search the list of available software packages, manage repositories, or upgrade the whole system.
Applications

Wine is a free and open source software application that aims to allow computer programs written for Microsoft Windows to run on Unix-like operating systems. Wine also provides a software library, known as Winelib, against which developers can compile Windows applications to help port them to Unix-like systems.
Window managers and desktop environments

A window manager is a type of software which handles windows, providing the user with the ability to control programs using the graphical user interface.

Some examples of window managers for UNIX-like systems are KWin, Sawfish, BlackBox, IceWM, Metacity and Afterstep.

A desktop environment includes a window manager and builds upon it. There are two main desktop environments which are used for UNIX-like systems: GNOME and KDE.
GNOME or KDE?

KDE and GNOME are complete desktop environments that consist of a large number of pieces of software. GNOME uses a window manager called Metacity and KDE uses KWin, but both of these desktops can be used with any other window manager.

These desktop environments can be customized by the user in order for them to meet his needs. However, some users may find one of them to be more useful for the type of activities they wish to undertake.

In December 2005, Linus Torvalds explicitly recommended the use of KDE because of it being easily configurable. Despite it aiming to be more user-friendly, GNOME still has options for power user hidden away in GConf (a system used by the GNOME desktop environment for storing configuration settings for the desktop and applications).
Differences in usability and performance

GNOME has less options readily visible. Its minimalistic offering of configuring options appeals to new users. Startup times are generally faster on GNOME desktop environments and so are standard actions, such as opening applications, browsing devices, loading icons etc.

KDE has more options readily visible. It is easily configurable but it can baffle new users. Besides occupying more memory, loading time on KDE desktop environments is longer because they have to load more from disk. A compensation for this, however, is the fact that once the libraries are loaded, applications start quicker.
Program equivalents Windows - Linux

The majority of the Windows programs are made on principle "all-in-one" (each developer adds everything to his or her product). This principle is called the "Windows-way". The ideology of UNIX/Linux is that one component or one program must execute only one task, but execute it well. ("UNIX-way"). The programs under Linux can be thought of as being like the LEGO building blocks.

This principle is very important and it is necessary to know it while searching for analogs of Windows-programs in Linux.
One of the alternatives for the graphics editing program Adobe Photoshop is GIMP (GNU image manipulation program). GIMP has a simple and easy to use Graphical User Interface compared to a more cluttered working environment in Photoshop, is faster and requires less disk space.

Professional photographers prefer Photoshop over GIMP, while GIMP is best for hobbyist or amateurs as a free alternative to Photoshop or as a learning tool before buying the expensive Adobe.
Microsoft Office was one of the first office suites of desktop applications: user-friendly, powerful and with new and unique capabilities. But over time, an open-source alternative, OpenOffice, has become increasingly more popular, making it a viable, affordable option for organizations seeking a productivity suite.
OpenOffice has a simple design which doesn't affect the power and functionality of the software but gives the advantage of being more usable and productive. Their performance is fairly similar, but Microsoft offers more advanced features in editing and document-viewing. Also, Microsoft offers quick and efficient support, in contrast to OpenOffice which has useful and extensive user and developer forums and a good deal of support across the net but isn't comparable to what is on offer for the other package.
Chapter IV
Connecting to Linux
About Hardware

- Computer hardware is a collection of physical elements that make up a usable PC.
- In today’s world, there is vast amount of operating systems and hardware in circulation.
- The command on Debian that allows the user to see all the hardware or peripheral components is “lspci“.
- According to the Linux manual, the “lspci“ command is “a utility for displaying information about PCI (Peripheral Component Interconnect) buses in the system and devices connected to them“.
- The devices are listed on the Console as series of numbers (including their PID numbers) as shown in the image below.
<table>
<thead>
<tr>
<th>PCI ID</th>
<th>Works?</th>
<th>Vendor</th>
<th>Device</th>
<th>Driver</th>
<th>Kernel</th>
</tr>
</thead>
<tbody>
<tr>
<td>08062e10</td>
<td>Yes</td>
<td>Intel Corp.</td>
<td>4 Series Chipset DRAM Controller</td>
<td>intel-app</td>
<td></td>
</tr>
<tr>
<td>08062e11</td>
<td></td>
<td>Intel Corp.</td>
<td>4 Series Chipset PCI Express Root Port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08062e12</td>
<td>Yes</td>
<td>Intel Corp.</td>
<td>4 Series Chipset Integrated Graphics Controller</td>
<td>i915</td>
<td>v2.6.32</td>
</tr>
<tr>
<td>08062e14</td>
<td></td>
<td>Intel Corp.</td>
<td>4 Series Chipset HECI Controller</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08062e16</td>
<td></td>
<td>Intel Corp.</td>
<td>4 Series Chipset PT IDER Controller</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08062e17</td>
<td></td>
<td>Intel Corp.</td>
<td>4 Series Chipset Serial KT Controller</td>
<td></td>
<td></td>
</tr>
<tr>
<td>080610de</td>
<td>Yes</td>
<td>Intel Corp.</td>
<td>82567LM-3 Gigabit Network Connection</td>
<td>e1000e</td>
<td>v2.6.28</td>
</tr>
<tr>
<td>08063a67</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) USB UHCI Controller #4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08063a68</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) USB UHCI Controller #5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08063a69</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) USB UHCI Controller #6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08063a6c</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) USB2 EHCI Controller #2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08063a6e</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) HD Audio Controller</td>
<td>snd-hda-intel</td>
<td>v3.1.0</td>
</tr>
<tr>
<td>08063a70</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) PCI Express Port 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08063a78</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) PCI Express Port 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08063a64</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) USB UHCI Controller #1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08063a65</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) USB UHCI Controller #2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08063a66</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) USB UHCI Controller #3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08063a6a</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) USB2 EHCI Controller #1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>08063a4e</td>
<td>Yes</td>
<td>Intel Corp.</td>
<td>82801 PCI Bridge</td>
<td>i810rng</td>
<td></td>
</tr>
<tr>
<td>08063a14</td>
<td></td>
<td>Intel Corp.</td>
<td>82801DO (ICH10DO) LPC Interface Controller</td>
<td>iTCO_s6dt</td>
<td>v2.6.28</td>
</tr>
<tr>
<td>08063a00</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) 4-port SATA IDE Controller</td>
<td>ata_pikx</td>
<td>v2.6.25</td>
</tr>
<tr>
<td>08063a60</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) SMBus Controller</td>
<td>i2c-801</td>
<td>v2.6.25</td>
</tr>
<tr>
<td>08063a06</td>
<td></td>
<td>Intel Corp.</td>
<td>82801JD/DO (ICH10 Family) 2-port SATA IDE Controller</td>
<td>ata_pikx</td>
<td>v2.6.25</td>
</tr>
</tbody>
</table>

You can help us!

Could you give me more information about your machine? Your information (PCI list and text you write below) is used only for making our HCL database better.
Connecting a device (e.g. printer) to the PC

In order to install a printer, one must at first install the „cups“ program using the Console.

After the printer is connected to the computer, the option of installing it can be found in the System Settings --> Printer.
Hardware compatibility and recognition on Linux

A HCL (Hardware Compatibility List) is a database of hardware models and their compatibility to certain operating systems.
Linux Hardware Compatibility List

• Mobile Phones: OpenMoko, Google Android
• Tablets and UMPCs: NoahPad UMPC, Nokia n810
• Laptops: Asus EeePC, OLPC XO
• Desktop PCs: Everex gPC, Dell Linux PCs
• Webcams etc.
The Android system is probably the most successful Linux-based device which became, by November 2011, the best-selling platform worldwide with over 200 million Android devices in use.

Android's kernel is based on the Linux kernel 2.6 and Linux Kernel 3.x, but has some architectural changes by Google such as autosleep and wakeclocks.
The kernel introduction

The word “kernel” makes someone probably think of the usually edible part of a seed or the most important part of an idea. When talking about computers, it is the connection between the hardware and the software, the central module of an operating system.
Our meeting with the kernel

Our first meeting with it was when we started the booting process. It is the one loaded first by the bootloader and remains in the main memory while being executed. That’s why, it is important to be as small as possible, but still capable of providing the essential services required by the rest of the operating systems and the applications. After that, it hands control to a special process called init, “the mother of all processes”. In addition, it creates two types of virtual files, which can be found in the directories /dev and /proc.
The kernel code is loaded into a protected area of memory, the kernel space and the user space being separated in order to prevent data from interfering. In Linux, it becomes a shell where the user can write commands to control the applications, make files/directories, manage everything and therefore use the operating system installed.
Kernel's functions

If we try to observe what a kernel does, you can say it has different functions: it plays the part of a **scheduler**, allocating every process the time needed, a **supervisor** who grants them the permission when scheduled, it **handles the problems**, taking care of the processes’ requests and interrupting them before anything bad happens and last but not least, it is a **memory manager** who redirects a certain part of it to every process in order to run.
Types of a kernel

There are more types of kernels, the most important being:

• **monolithic**, usually used by the Unix-like operating systems (such as Linux, FreeBSD),

• **hybrid** (Microsoft Windows NT, 2000, XP, Vista, 7 and MAC OS X),

• **microkernels** (MINIX, GNU Hurd)

• **exokernels**, which are very small and still experimental

The main differences between the first three categories mentioned are: monolithic kernels can load modules on their own, microkernels include some additional code in the kernel space to increase performance and the hybrid ones are made to run swiftly, but slower.
Because Linux is a monolithic kernel, the kernel can use modules. These modules are pieces of code that can be loaded and unloaded when needed for different features requested by the user, but without having to reboot the system. They can be considered an extension of the kernel itself with the same functionality. Therefore, it doesn't have to load everything in the booting process.

For example, one type of a module can be a device driver, which allows the hardware to connect to the system.
While the kernel is the heart of Linux, a lot of people don't know much about the Windows one, mostly because it's hidden and has a closed source. The Linux kernel is faster and can encounter fewer bugs, but because the modules run in the same address space, one of these bugs can bring down the whole system.
Chapter V

Practical uses and file extensions
In 3 weeks we got used with the console and now we don't have problems using commands and handling files and directories.
ted@debianTED:~$ pwd
/home/ted

123456789  authorized_keys  Desktop  Downloads  index.html  Pictures  set  Templates  test2  www
apache2.conf  BLA  BLA  Documents  fstab  Music  Public  temp  test1  Videos
ted@debianTED:~$ ls
directory/
ted@debianTED:~$ cd directory/
ted@debianTED:~$ touch file1
ted@debianTED:~$ ls
directory/
ted@debianTED:~$ touch file2
This is a text document.
ted@debianTED:~$ rm file2
ted@debianTED:~$ cd ..
ted@debianTED:~$ cd directory/
ted@debianTED:~$ cd ~
ted@debianTED:~$ w
08:57:18 up 34 min, 1 user, load average: 0.94, 0.81, 0.70

USER      TTY            FROM     LOGIN@   IDLE   JCPU  PCPU   WHAT
ted        pts/0       :0       08:35   0.00s  0.10s  0.00s  w

ted@debianTED:~$ whoami
ted

ted@debianTED:~$ su
Password:
root@debianTED:/home/ted# passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@debianTED:/home/ted# whoami
root
root@debianTED:/home/ted# w
08:58:43 up 36 min, 1 user, load average: 0.37, 0.67, 0.66

root@debianTED:/home/ted#
root@debianTED:/home/ted# ls
123456789  aaa.tar  authorized_keys  Desktop  Documents  fstab  Music  Public  temp  test1  Videos
directory  Downloads  index.html  Pictures  set  Templates  test2  www
root@debianTED:/home/ted# cd directory/
root@debianTED:/home/ted/directory# ls
file1  file2  file3  file4  file5  file6
root@debianTED:/home/ted/directory# cd ..
root@debianTED:/home/ted#/tar cvf archive.tar directory
root@debianTED:/home/ted# rm -r directory
root@debianTED:/home/ted# tar tvf archive.tar
drwxr-xr-x root/root  0 2012-09-12 09:39 directory/
-rw-r--r-- root/root  0 2012-09-12 09:39 directory/file4
-rw-r--r-- root/root  0 2012-09-12 09:39 directory/file6
-rw-r--r-- root/root  0 2012-09-12 09:39 directory/file3
-rw-r--r-- root/root  0 2012-09-12 09:39 directory/file2
-rw-r--r-- root/root  0 2012-09-12 09:39 directory/file1
-rw-r--r-- root/root  0 2012-09-12 09:39 directory/file5
root@debianTED:/home/ted#/tar xvf archive.tar
directory/
directory/file4
directory/file6
directory/file3
directory/file2
directory/file1
directory/file5
root@debianTED:/home/ted#/ls
123456789  aaa.tar  archive.tar  directory  BLA  directory  Downloads  index.html  Pictures  set  Templates  test2  www
directory  Documents  fstab  Music  Public  temp  test1  Videos
root@debianTED:/home/ted#/cd directory/
root@debianTED:/home/ted/directory# ls
file1  file2  file3  file4  file5  file6
NAME
ps - report a snapshot of the current processes.

SYNOPSIS
ps [options]

DESCRIPTION
ps displays information about a selection of the active processes. If you want a repetitive update of the selection and the displayed information, use `top(1)` instead.

This version of ps accepts several kinds of options:

1. UNIX options, which may be grouped and must be preceded by a dash.
2. BSD options, which may be grouped and must not be used with a dash.
3. GNU long options, which are preceded by two dashes.

Options of different types may be freely mixed, but conflicts can appear. There are some synonymous options, which are functionally identical, due to the many standards and ps implementations that this ps is compatible with.

Note that "ps -aux" is distinct from "ps aux". The POSIX and UNIX standards require that "ps -aux" print all processes owned by a user named "x", as well as printing all processes that would be selected by the -a option. If the user named "x" does not exist, this ps may interpret the command as "ps aux" instead and print a warning. This behavior is intended to aid in transitioning old scripts and habits. It is fragile, subject to change, and thus should not be relied upon.

By default, ps selects all processes with the same effective user ID (euid=EUID) as the current user and associated with the same terminal as the invoker. It displays the process ID (pid=PID), the terminal associated with the process (tname=TTY), the cumulated CPU time in [DD-]hh:mm:ss format (time=TIME), and the executable name (cmd=CMD). Output is unsorted by default.

The use of BSD-style options will add process state (stat=STAT) to the default display and show the command args (args=COMMAND) instead of the executable name. You can override this with the PS_FORMAT environment variable. The use of BSD-style options will also change the process selection to include processes on other terminals (TTYs) that are owned by you; alternately, this may be described as setting the selection to be the set of all processes filtered to exclude processes owned by other users or not on a terminal. These effects are not considered when options are described as being "identical" below, so -M will be considered identical to Z and so on.

Except as described below, process selection options are additive. The default selection is discarded, and then the selected processes are added to the set of processes to be displayed. A process will thus be shown if it meets any of the given selection criteria.

EXAMPLES
To see every process on the system using standard syntax:
```ps -e
ps -ef
ps -Ef
ps -esy```

To see every process on the system using BSD syntax:
```ps ax
ps axu```
root@debianTED:~/# history
● Now it's time to write a shell script and see how it works...

If you don't want to waste time typing a frequently used sequence of commands at a command prompt, more especially if they are abnormally long or complex, you should consider making a script. Scripting is a way by which one can alleviate this necessity by automating these command sequences in order to make ones life at the shell easier and more productive. Scripting is all about making the computer, the tool, do the work.
clear
sleep 2
echo "Hey!"
echo "I am "
whoami
sleep 5
echo "We are on"
date +DATE:%d%m%y
echo "and the time is"
date +TIME:%H%M
sleep 10
echo "Hm, I'm bored. Let's execute an update and an upgrade."
sleep 5
aptitude update SS aptitude upgrade
sleep 10
echo "My job here is done"
sleep 3
clear
echo "Bye!"
Hey!
I am root
We are on
DATE: 130912
and the time is
TIME: 1047
Hmm, I'm bored. Let's execute an update and an upgrade.
Get: 1 http://security.debian.org wheezy/updates InRelease [87.8 kB]
Get: 2 http://security.debian.org wheezy/updates/main Sources [937 B]
Get: 3 http://ftp.de.debian.org wheezy InRelease [207 kB]
Get: 4 http://security.debian.org wheezy/updates/main amd64 Packages [979 B]
Get: 5 http://ftp.de.debian.org wheezy/main Sources DiffIndex [7,876 B]
Get: 6 http://ftp.de.debian.org wheezy/main amd64 Packages DiffIndex [7,876 B]

Ign http://security.debian.org wheezy/updates/main Translation-sk
Ign http://security.debian.org wheezy/updates/main Translation-km
Ign http://security.debian.org wheezy/updates/main Translation-nb
Ign http://security.debian.org wheezy/updates/main Translation-id
Ign http://security.debian.org wheezy/updates/main Translation-cs
Ign http://security.debian.org wheezy/updates/main Translation-ca
Ign http://security.debian.org wheezy/updates/main Translation-fr
Ign http://security.debian.org wheezy/updates/main Translation-sr
Ign http://security.debian.org wheezy/updates/main Translation-eo

Fetchedy 453 kB in 35s (12.8 kB/s)

No packages will be installed, upgraded, or removed.
0 packages upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 0 B of archives. After unpacking 0 B will be used.

My job here is done
File extensions

These are suffixes used to indicate the encoding of content or usage.

- .tar - archived
- .tar.gz - archived then compressed
- .txt - a plain ASCII text file
- .conf - configuration file
- .pdf
- .sh - shell script
- .jpg - a graphical or image file
- .html / .htm - an HTML file
- .src - a source code file written in plain text; a source must be compiled to be used
- .iso - a image(copy) of a CD-ROM or DVD file
- .odf + .odt + .odp - they are file extensions for libreoffice package; just like .ppt or .docx for Microsoft Office
Chapter VI

Facing Linux
http://youtu.be/_tMv6zBOEpo
First Impressions (first week)

- When we first entered the room, there was the big screen in front of us and we thought about how simple it was just to follow the commands on it.

- Then we received the printed pages with the first installation commands and we thought it was going to be completely different from what it proved to be in the end.
And then we started to wait for all those programs to install and update and upgrade and we realized that we should not only copy those commands, but also understand them, not only type them, but also find out why they help us work with Linux.
During the second week, we began to understand the structure of a command and how it works. We made our first steps into the process of understand this new system of operation.

It was very useful for us that we could ask for help every time we had an error.
The last week of the project made us realize how useful and advantageous working with Linux really is. We should have been more open-minded in the previous days and not always consider Windows better than Linux just because we were much more familiar with the latter one. We also found out the complexity of computer usage.
While studying Linux, we discovered the way in which we can enjoy working with the computer. For example, an interesting thing we have done was getting into the other’s computers with ssh command. This command helped us understand the similarities and differences between programs used with Widows (TeamViewer for example) and their equivalents in Linux (ssh).

```
File Edit View Search Terminal Help
iulia@IULIA:~$ ssh placinta@192.168.0.105
The authenticity of host '192.168.0.105 (192.168.0.105)' can't be established.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or 'no': yes
Warning: Permanently added '192.168.0.105' (ECDSA) to the list of known hosts.
placinta@192.168.0.105's password:
Linux Teodora 3.2.0-3-amd64 #1 SMP Mon Jul 23 02:45:17 UTC 2012 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Sep  5 12:07:57 2012 from iulia.local
placinta@Teodora:~$ `
First!!

We're in Germany!

Creating Our First CMS

Really nice experience, taking into account the fact that everyone in our team has a different opinion on the site's color scheme.
Opinions on the Project

Theodor Ivan: “I actually thought it would be much more difficult as it really was. I discovered it is quite easy because of the friendly interface and because of the fact that the commands can be easily learnt. I find it useful that now I am able to use two different operating systems!”

Bianca Miron: “In my opinion, it was a really interesting experience. Learning how to use something I’ve never seen before was hard at the beginning, but taking it step by step, it became accessible. I’ll look forward to my second meeting with Linux!”
We consider this project a real challenge because there are so many things to learn in such a short period of time. All in all, it was a very interesting experience which got the best out of us. We are grateful for having studied with a very good teacher who helped us understand Linux!