DevOps external course

Linux Essentials. Lection 1

Lecture 5.1

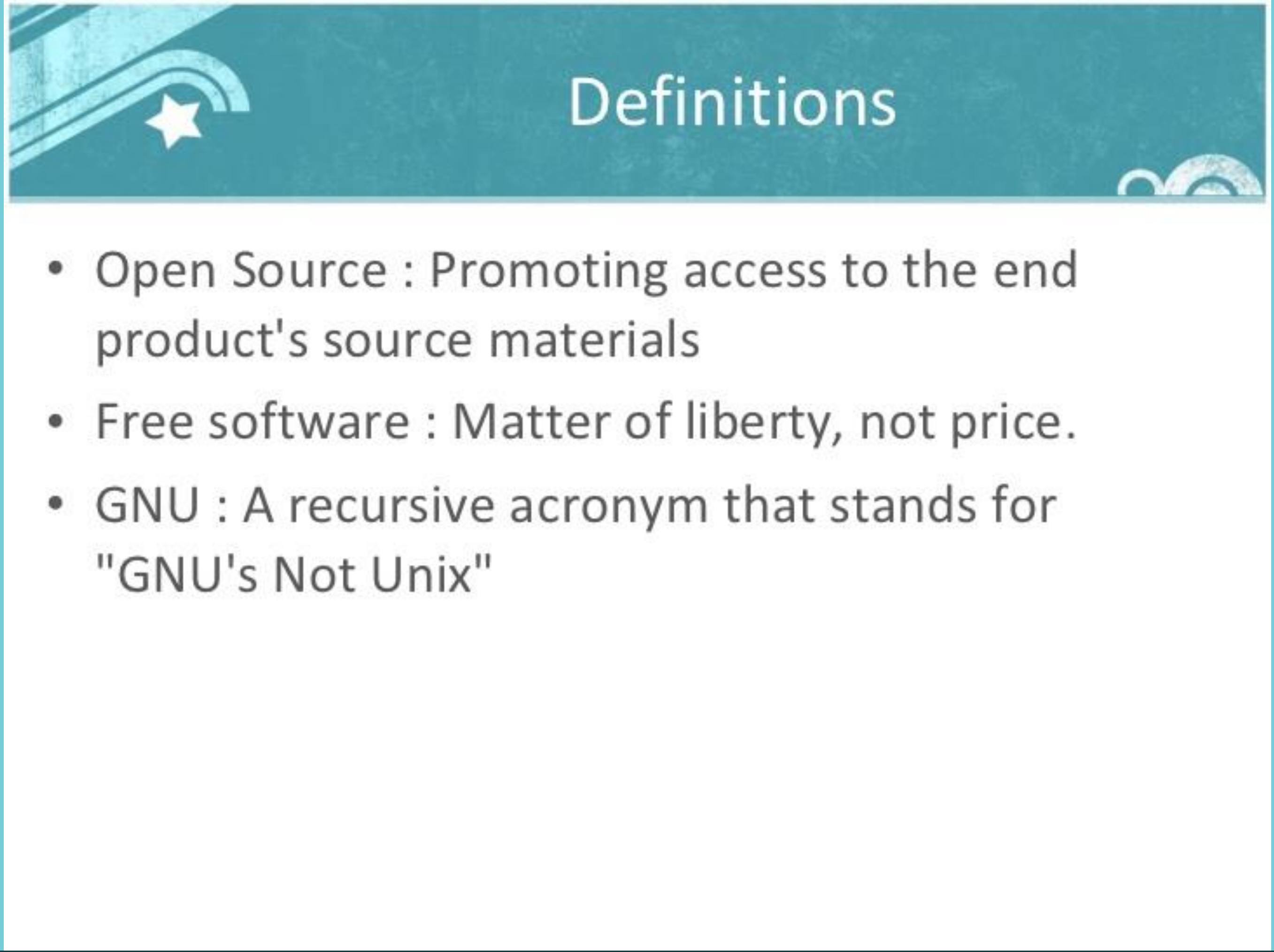
Module 5 **Linux Essentials**

Serge Prykhodchenko

Agenda

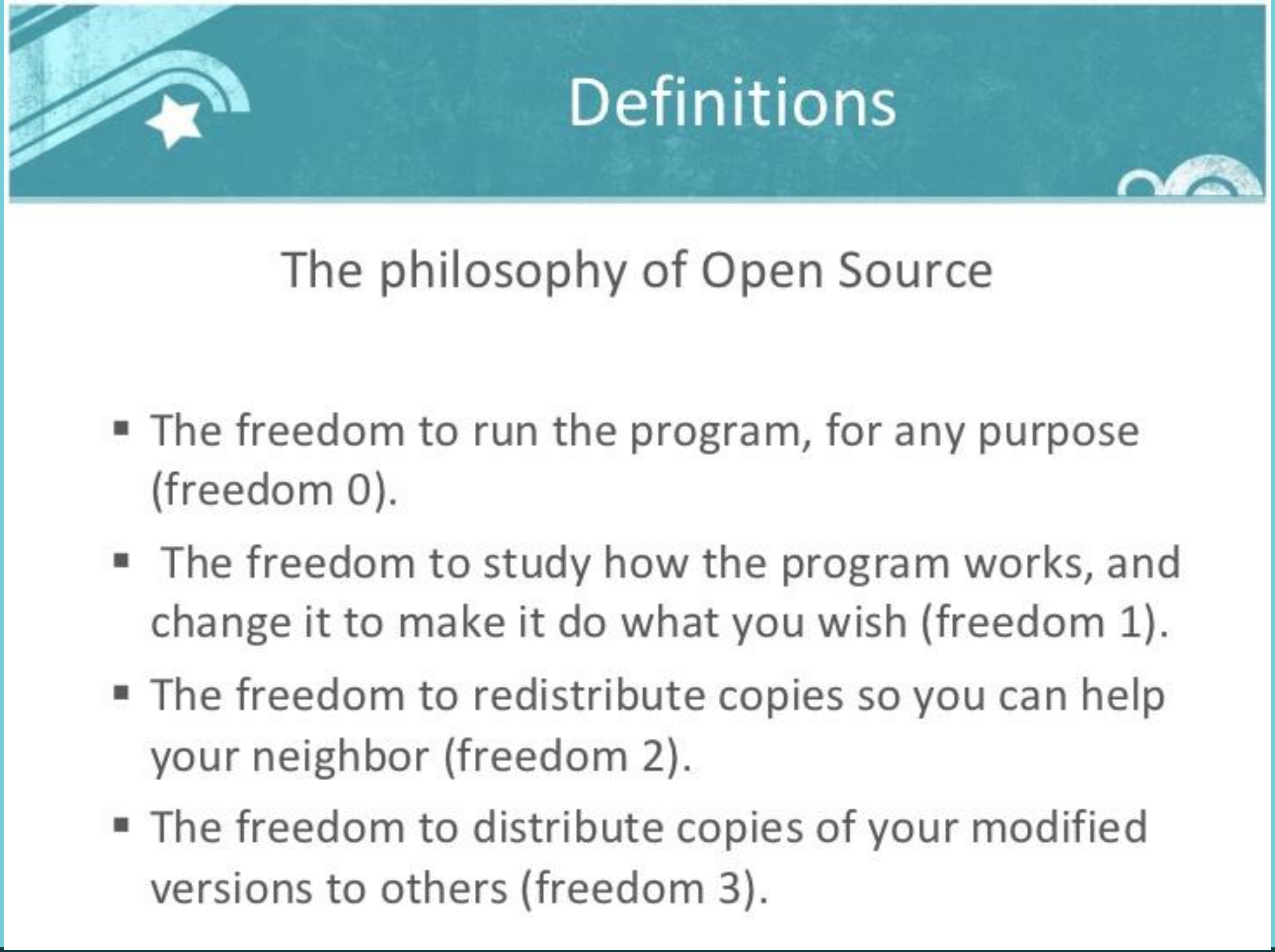
- Open Source Software
- OS
- Linux
- First commands
- Embedded
- Q&A

OPEN SOURCE SOFTWARE



Definitions

- Open Source : Promoting access to the end product's source materials
- Free software : Matter of liberty, not price.
- GNU : A recursive acronym that stands for "GNU's Not Unix"

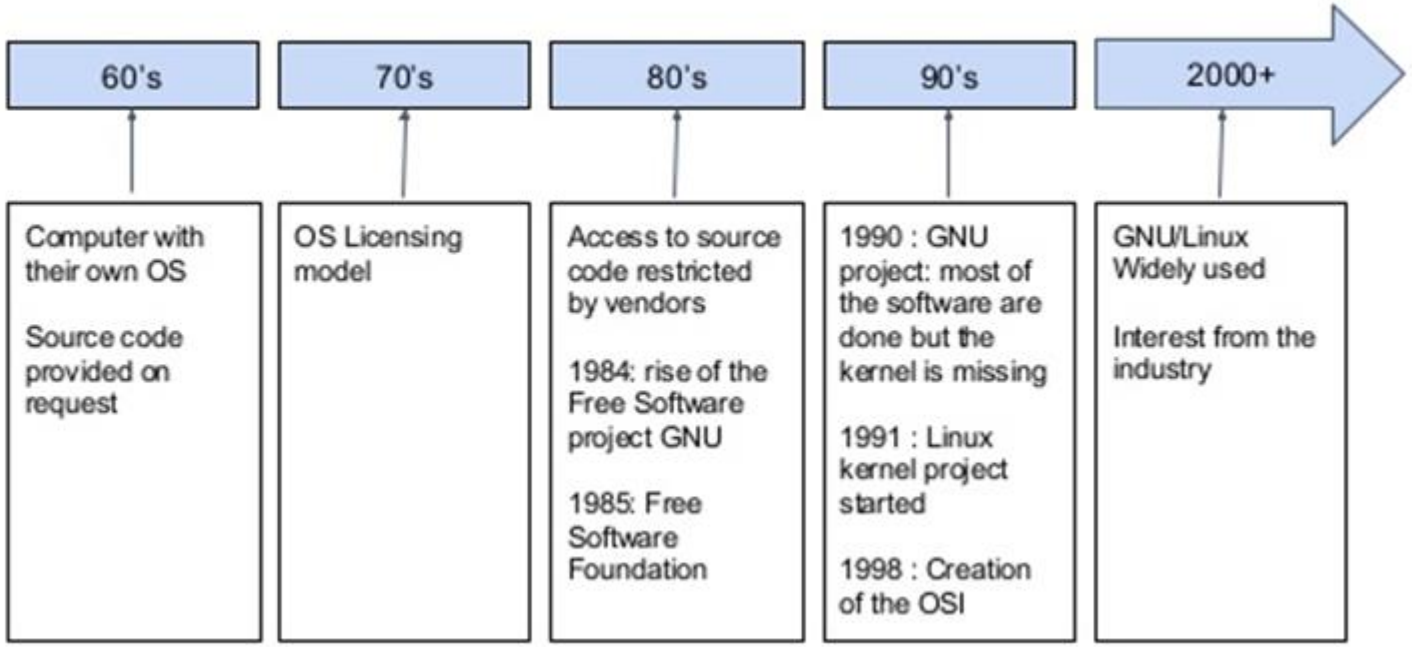


Definitions

The philosophy of Open Source

- The freedom to run the program, for any purpose (freedom 0).
- The freedom to study how the program works, and change it to make it do what you wish (freedom 1).
- The freedom to redistribute copies so you can help your neighbor (freedom 2).
- The freedom to distribute copies of your modified versions to others (freedom 3).

Open Source Origin



60's

Computer with their own OS

Source code provided on request

70's

OS Licensing model

80's

Access to source code restricted by vendors

1984: rise of the Free Software project GNU

1985: Free Software Foundation

90's

1990 : GNU project: most of the software are done but the kernel is missing

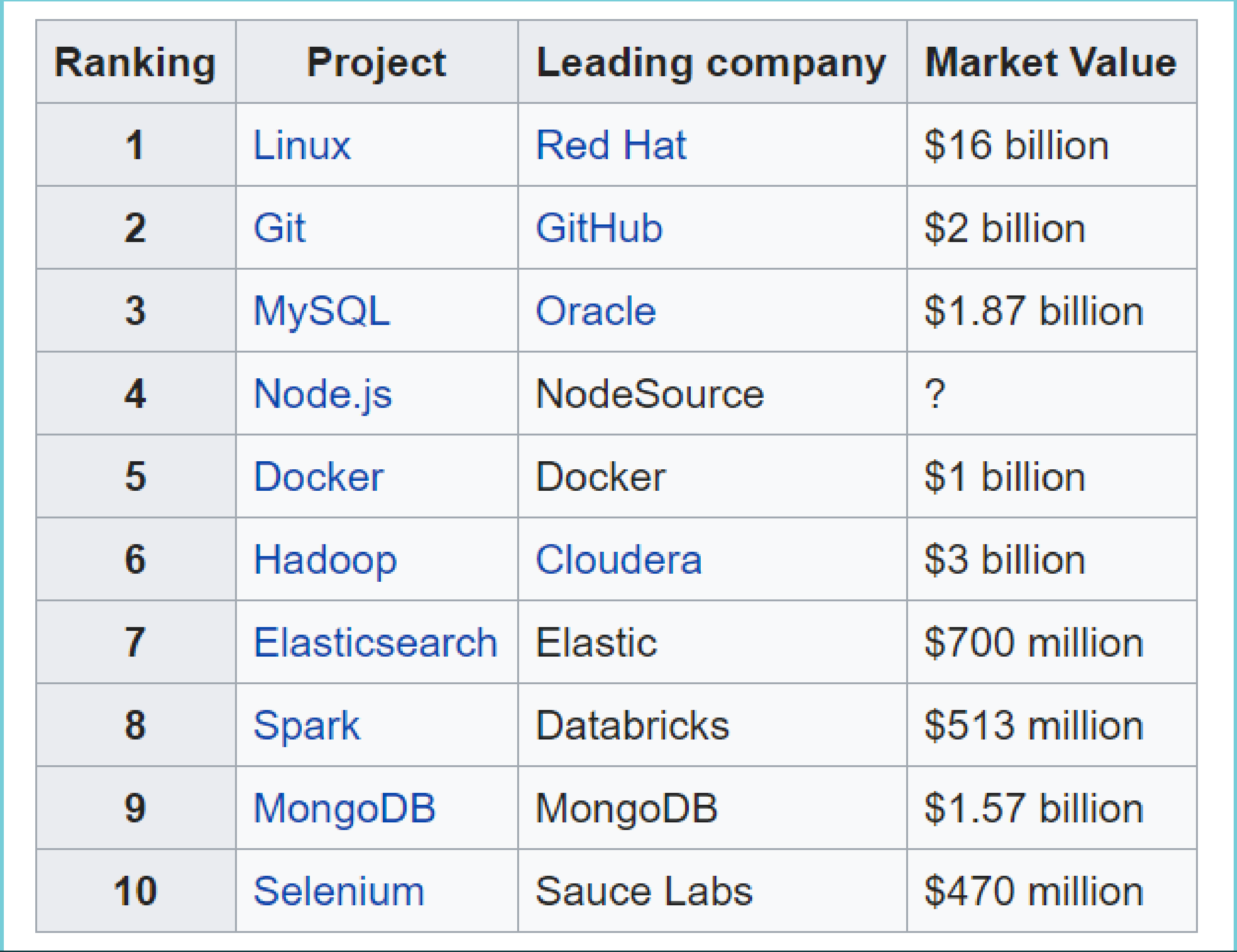
1991 : Linux kernel project started

1998 : Creation of the OSI

2000+

GNU/Linux Widely used

Interest from the industry



Ranking	Project	Leading company	Market Value
1	Linux	Red Hat	$16 billion
2	Git	GitHub	$2 billion
3	MySQL	Oracle	$1.87 billion
4	Node.js	NodeSource	?
5	Docker	Docker	$1 billion
6	Hadoop	Cloudera	$3 billion
7	Elasticsearch	Elastic	$700 million
8	Spark	Databricks	$513 million
9	MongoDB	MongoDB	$1.57 billion
10	Selenium	Sauce Labs	$470 million

With Open Source Software you can:

- Get access to the source code.
- Permission to change the software.
- Free distribution of original and modified code.
- Having derived work that can be distributed under the same terms of original software.
- The same license of the original software. You can take a new license, but it is not necessary.
- Sometimes, If you use it, it is not necessary that your program has to be open source too.
- The principle here is to promote the collaboration inside a community to generate mutual benefits.
- Not all software open source is necessarily free. And free software can be also open source at the same time.

With free software you can:

- Use the software.
- Run it.
- Understand who it works.
- Share and distribute it.
- Create another software only if you respect these aspects.
- And what you think? Software for commercial use also can be free software. Why? Well, if you respect all these points, you can charge a rate for distribution.

Proprietary software meaning:

- Software that you have to buy if you want to use it.
- This software belongs to someone else, but what does it mean? The code is closed, it is copyrighted, its use is limited at some point, especially when it is referred to distribution or modification.
- Proprietary software also is called commercial software or closed-source software inattention to one of its most important characteristics.
- This software also is really good and unique and sometimes, they can be modified within creators limits.
- Some of the most important software programs that revolutionized the world years ago were in this category.

Characteristic of proprietary software

- It has to be bought
- Has a license which is the property of a developer, company or the owner.
- Without access to its source code
- Free distribution or copy is prohibited. Actually, it is a crime
- Its use depends on the end-users agreement
- They can take you to jail if you violate any rule or agreement you accepted before.

GNU not Unix

- GNU project
  - Established in 1984 by Richard Stallman, who believes that software should be free from restrictions against copying or modification in order to make better and efficient computer programs
  - GNU is a recursive acronym for “GNU's Not Unix”
  - Aim at developing a complete Unix-like operating system which is free for copying and modification
  - Companies make their money by maintaining and distributing the software, e.g. optimally packaging the software with different tools (Redhat, Slackware, Mandrake, SuSE, etc)
  - Stallman built the first free GNU C Compiler in 1991. But still, an OS was yet to be developed

Free Software Foundation

- The Free Software Foundation (FSF) is a non-profit organization founded by Richard Stallman on 4 October 1985 to support the free software movement, which promotes the universal freedom to study, distribute, create, and modify computer software, with the organization's preference for software being distributed under copyleft ("share alike") terms, such as with its own GNU General Public License. The FSF was incorporated in Boston, Massachusetts, US, where it is also based.
- From its founding until the mid-1990s, FSF's funds were mostly used to employ software developers to write free software for the GNU Project. Since the mid-1990s, the FSF's employees and volunteers have mostly worked on legal and structural issues for the free software movement and the free software community.

GNU General Public License

- The GNU General Public License (GNU GPL or GPL) is a series of widely used free software licenses that guarantee end users the freedom to run, study, share, and modify the software.[7] The licenses were originally written by Richard Stallman, former head of the Free Software Foundation (FSF), for the GNU Project, and grant the recipients of a computer program the rights of the Free Software Definition.[8] The GPL series are all copyleft licenses, which means that any derivative work must be distributed under the same or equivalent license terms. This is in distinction to permissive software licenses, of which the BSD licenses and the MIT License are widely-used less-restrictive examples. GPL was the first copyleft license for general use.

GNU Lesser General Public License

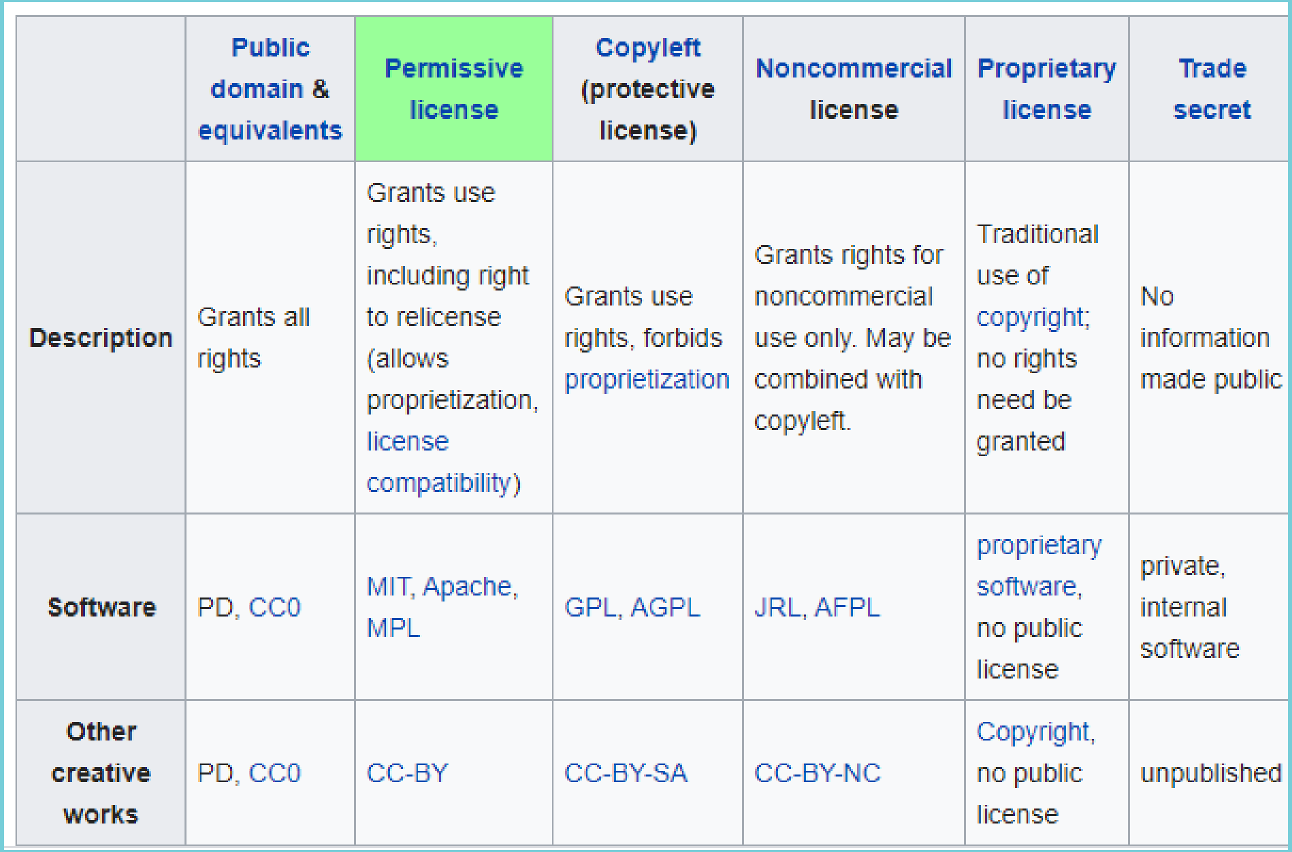
The GNU Lesser General Public License (LGPL) is a free-software license published by the Free Software Foundation (FSF). The license allows developers and companies to use and integrate a software component released under the LGPL into their own (even proprietary) software without being required by the terms of a strong copyleft license to release the source code of their own components. However, any developer who modifies an LGPL-covered component is required to make their modified version available under the same LGPL license. For proprietary software, code under the LGPL is usually used in the form of a shared library, so that there is a clear separation between the proprietary and LGPL components. The LGPL is primarily used for software libraries, although it is also used by some stand-alone applications.

The LGPL was developed as a compromise between the strong copyleft of the GNU General Public License (GPL) and more permissive licenses such as the BSD licenses and the MIT License. The word "Lesser" in the title shows that the LGPL does not guarantee the end user's complete freedom in the use of software; it only guarantees the freedom of modification for components licensed under the LGPL, but not for any proprietary components.

BSD licenses

- BSD licenses are a family of permissive free software licenses, imposing minimal restrictions on the use and distribution of covered software. This is in contrast to copyleft licenses, which have share-alike requirements. The original BSD license was used for its namesake, the Berkeley Software Distribution (BSD), a Unix-like operating system. The original version has since been revised, and its descendants are referred to as modified BSD licenses.
- BSD is both a license and a class of license (generally referred to as BSD-like). The modified BSD license (in wide use today) is very similar to the license originally used for the BSD version of Unix. The BSD license is a simple license that merely requires that all code retain the BSD license notice if redistributed in source code format, or reproduce the notice if redistributed in binary format. The BSD license (unlike some other licenses) does not require that source code be distributed at all.

Types of licenses



	Public domain & equivalents	Permissive license	Copyleft (protective license)	Noncommercial license	Proprietary license	Trade secret
Description	Grants all rights	Grants use rights, including right to relicense (allows proprietization, license compatibility)	Grants use rights, forbids proprietization	Grants rights for noncommercial use only. May be combined with copyleft.	Traditional use of copyright; no rights need be granted	No information made public
Software	PD, CC0	MIT, Apache, MPL	GPL, AGPL	JRL, AFPL	proprietary software, no public license	private, internal software
Other creative works	PD, CC0	CC-BY	CC-BY-SA	CC-BY-NC	Copyright, no public license	unpublished

Software open source vs proprietary software: advantages and disadvantages

Software open source Advantages

- You can adapt it to your necessities even from source code.
- All replica or distribution it is possible although you haven’t paid it.
- Free support because the same community that uses the software, frequently tend to answer questions, giving advice, making forums and provide detail documentation.
- Fewer errors and faster solutions. This is related to the previous point. Projects with open source literally could have millions of people looking it, using it, and getting better.
- For that reason, some experts think that open source software is safer.
- It is universal.

Software open source Disadvantages

- Limited warranty. This happens because lots of people can change it. Also usually they haven’t liability or infringement indemnity protection.
- Open source software can have compatibility issues, and solving it could cost a lot of money.

Software open source vs proprietary software: advantages and disadvantages

Proprietary Software Advantages

- Stability. This maybe is the most important advantage. Creator gives you a software which it was probed and it is capable to do perfectly all things an actions you saw before buying it.
- Reliability and warranty of 100% from creators.
- Proprietary software is unique. You won’t find it in any place different from the provider.
- Most compatibility in some cases.

Proprietary Software Disadvantages

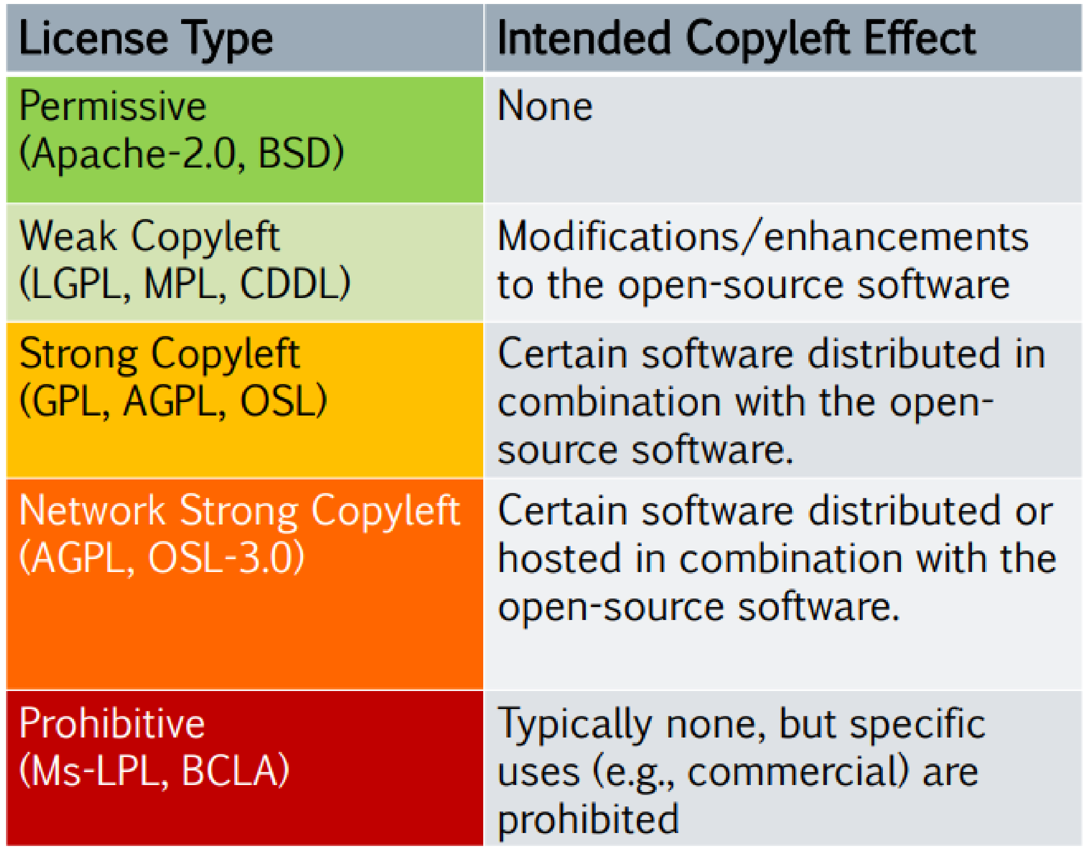
- Higher cost. But if you look at how an invest, it doesn’t matter much.
- You cannot modify the source code.
- You cannot share it or distribute it.
- You will be totally dependent on creators to upgrade and maintain the software in the source.
- Some specialist thinks that Proprietary software is less safe because security will depend on software producers.

Open-source Business Models

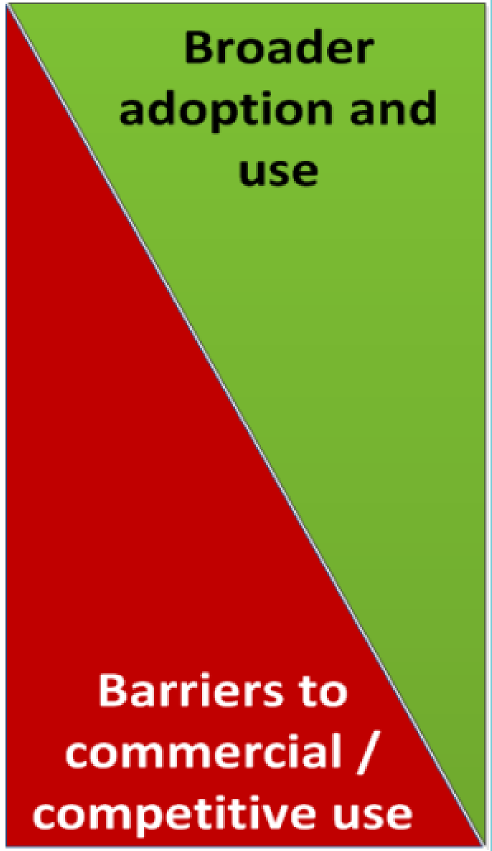
Open-source business models usually rely upon one or more of the following strategies:

1. Dual-licensing proprietary company software;
2. Providing commercial or enterprise versions, plugins, or extensions to open-source products;
3. Offering maintenance, support, consulting, or other services that support or complement open-source products;
4. Offering hosting, warranty, indemnity, or other products that complement open-source products; and
5. Closed-source modified distributions of open-source products.

Open-source License Types



License Type	Intended Copyleft Effect
Permissive (Apache-2.0, BSD)	None
Weak Copyleft (LGPL, MPL, CDDL)	Modifications/enhancements to the open-source software
Strong Copyleft (GPL, AGPL, OSL)	Certain software distributed in combination with the open-source software.
Network Strong Copyleft (AGPL, OSL-3.0)	Certain software distributed or hosted in combination with the open-source software.
Prohibitive (Ms-LPL, BCLA)	Typically none, but specific uses (e.g., commercial) are prohibited



OPERATING SYSTEM

Operating system

An operating system is a collection of programs, both conventional and firmware that provide an interface between user applications and computer hardware.

OS functions:

- define the so-called "user interface";
- ensure the sharing of hardware resources between users and / or applications;
- to give an opportunity to work with common data in the mode of collective use;
- plan user access to shared resources;
- to ensure efficient performance of input-output operations;
- to carry out the recovery of information and computing process in case of errors;
- provide an opportunity for software development;
- keep records of the use of resources.

Operating system

The operating system manages the following main resources:
- processors;
- memory;
- input-output devices;
- data.

During its operation, the OS interacts with:
- hardware;
- system programmers;
- application programmers;
- programs;
- users.

Hardware - the computer hardware itself.

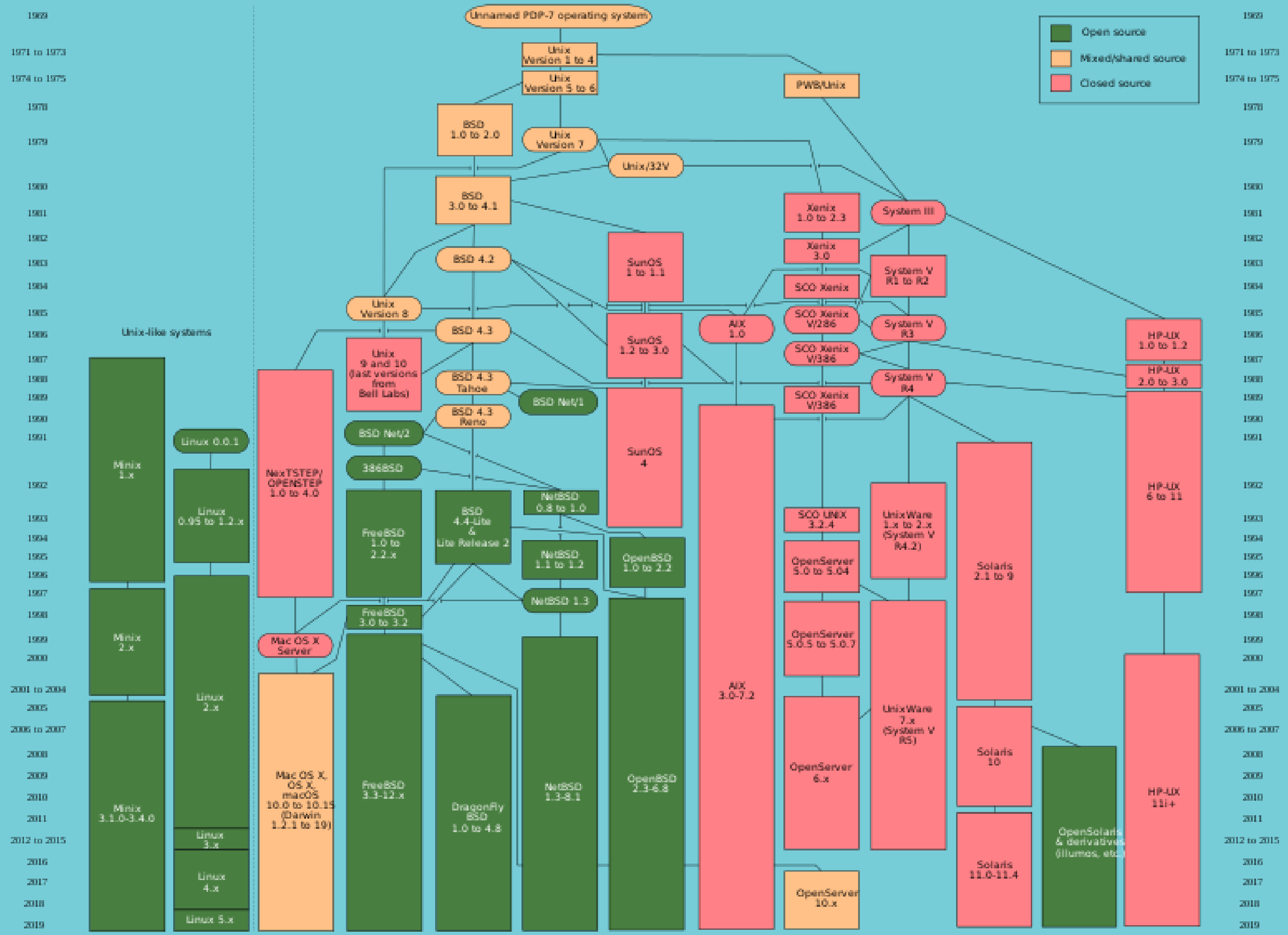
System programmers - usually involved in maintaining the operating system, writing device drivers and various system utilities, etc.

Application programmers - write programs that will be used by users to solve specific problems.

Users are subscribers of a computer system who use a computer to do useful work.

Programs access an operating systems using special commands, procedures and functions. This enables users to use the services provided by the operating system without compromising an integrity and performance.

The evolution of a Linux operation systems family.



https://en.wikipedia.org/wiki/File:Unix_history-simple.svg

What is Linux?

- First released in 1991 by a University of Finland student Linus Torvalds.
- Basically a kernel, it was combined with the various software and compilers from GNU Project to form an OS, called GNU/Linux
- Linux is a full-fledged OS available in the form of various Linux Distributions
- RedHat, Fedora, SuSE, Ubuntu, Debian are examples of Linux distros
- Linux is supported by big names as IBM, Google, Sun, Novell, Oracle, HP, Dell, and many more

What is a Linux Distribution?

- Linux Kernel
- Supporting features and programs
  - Usually tailored to a particular purpose
- With so many options, it is easy to find the perfect solution.

What are the Major Ones?

- **Red Hat**: One of the earliest players in the game, Red Hat now position itself strongly in the business market. It has created a community-supported distribution, Fedora Core, which is the choice of many for desktop use.
- **Debian**: The most popular community-created distribution. Debian is an excellent choice for server environments. Debian has also been used as the base for many specialist distributions.
- **Ubuntu**: Desktop usability, out of the box. Taglined "Linux for human beings,” Based on Debian.
- **SUSE**: Novell's answer to Red Hat, comes in “enterprise” and a community-based OpenSUSE

- All Distributions have their respective strengths.

THE MAIN CHARACTERISTICS OF LINUX OSes

Real multitasking. All processes are independent; none of them should interfere with other tasks. For this kernel carries out the CPU time-sharing mode alternately allocating time interval to execute.

Multi-user access. Linux is not only a multitasking OS, it supports the simultaneous operation of many users. In this case, Linux can use all system resources, working with the host through various remote terminals.

Swap RAM to disk. Swapping RAM to disk allows you to work with a limited amount of RAM; for this, the contents of some parts of the RAM are written to a dedicated area on the hard disk, which is treated as additional RAM. This slows down the speed of work a bit but allows you to organize the operation of programs that require more RAM than the computer has.

Memory paging. Linux system memory is organized in 4K pages. If the RAM is completely depleted, the OS will look for long-unused memory pages to move them from memory to the hard drive. Linux recovers them from disk.

Loading of executable modules "on demand". The Linux kernel supports on-demand paging, whereby only the necessary portion of the executable program code is located in memory, and the unused portions remain on disk.

THE MAIN CHARACTERISTICS OF LINUX OSes

Sharing executable programs. One copy of the executable code of this application is loaded into memory, which is used simultaneously by several executable tasks.

Shared library. Libraries are sets of procedures used by programs to process data. There are a number of standard libraries used by more than one industry at the same time.

100% POSIX 1003.1 compliant. Partial support for System V and BSD features POSIX 1003.1 (Portable Operating System Interface - Mobile Operating System Interface) defines a standard Unix system interface that is described by a set of C procedures. Linux is 100% POSIX compliant. Several System V and BSD features are additionally supported to increase compatibility.

Support for various file system formats. Linux supports a wide variety of file system formats, including DOS file systems, NTFS, and modern journaling file systems. At the same time, and which is its own Linux file system, called the Second Extended File System (ext2fs), allows efficient use of disk space.

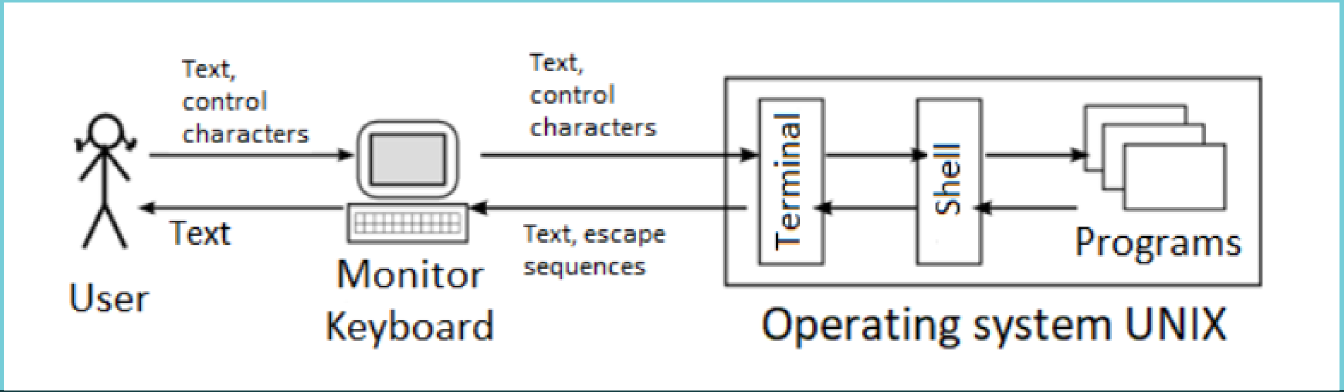
Networking capabilities. Linux can be integrated into any local network. All Unix services are supported, including network file system (NFS), remote access (telnet, rlogin, ssh), TCP / IP networking, remote access via SLIP and PPP, etc. It also supports turning on a Linux machine as server or client for another network, in particular file sharing (sharing) and remote printing on Macintosh, NetWare and Windows works.

THE MAIN CHARACTERISTICS OF LINUX OSes

A distribution kit is a set of software that includes all 4 main components of the OS, i.e. the kernel, file system, shell and set of utilities, as well as some set of application programs. Typically, all programs included in a Linux distribution are distributed under the GPL.

Using diskless stations. On the server computer (in fact, the server can be a regular Linux PC), special software is installed to boot a small Linux kernel (specially assembled to run on a morally obsolete computer) over the network, as well as a set of initialization scripts.

Terminal. In the UNIX operating system, the primary means of user interaction with the system are the keyboard and the text-mode monitor screen



References

- Wikipedia

  http://en.wikipedia.org/wiki/SUSE_Linux

  http://en.wikipedia.org/wiki/Red_Hat_Linux

  http://en.wikipedia.org/wiki/Ubuntu_%28Linux_distribution%29

  http://en.wikipedia.org/wiki/Yellow_Dog_Linux

- DistroWatch

http://distrowatch.com/

https://www.ibiblio.org/software/distributions/

THE MAIN CHARACTERISTICS OF LINUX OSes

Terminal login procedure. As mentioned earlier, you must provide a unique name (account) and password to log in.

Account (account) - a system object with which Linux keeps track of the user's work in the system. The account contains the user data needs for registration in the system and further work with it.

Login name - the name of the user account that must be entered when registering in the system.

Home directory. in Linux all users files are stored separately, each user has his own home directory in which he can store his data. Other users' access to the user's home directory may be restricted. Information about the home directory must be in the account, because the registrated user start work with it.

Often, the user's **home directory** is /home/username, but the administrator is free to customize it.

THE MAIN CHARACTERISTICS OF LINUX OSes

Command shell. Each user needs to provide a way to interact with the system: sending commands to system and getting responses from it. For this used, a special program - the command shell (or command line interpreter). It must be launched for every user who logs on to the system.On Linux are available different command line interpreters,in account you have a information which one to run for each users
If you do not specify a command shell when creating an account, it will be default, most likely it will be bash.

A command line interpreter (command interpreter, command shell, shell, CLI) its a program, which Linux used to organize a "dialogue" between a person and a system. The command interpreter has three main components: (1) a command line editor and analyzer, (2) a high-level system-oriented programming language, (3) a means of organizing the interaction of commands with each other and with the system.

Command line syntax. Most of all, communication in this language resembles a written dialogue with the system - an alternate exchange of texts. The user's statement in this language is a command, each command is a separate line. Until enter is pressed, the line can be edited, then it will be send to the shell. The shell parses the received command - translates it into the language of system objects and functions, and then sends it to the system for execution.

THE MAIN CHARACTERISTICS OF LINUX OSes

Command history. The Bash shell provides to the user command line tools to manage command history. Command history is, first of all, a very handy tool that shortens manual input.

For command history, you can use the Bash command: ***history***

```
[globus @ fedora ~] $ history
………
365 history | less
366 history
367 echo $ HISTCMD
368 echo $ HISTFILE
369 history
[globus @ fedora ~] $
```

To view the commands history, are often used the keys ↑ ↓ - which allow you to navigate in commands history.

Completion mechanism. Abbreviations allow you to quickly type commands, and file names, which most often are parameters of these commands. It happens when typed line - the path to the file and the first few letters of its name - is enough to unambiguously point to this file, because there are no more files along the entered path, whose name begins with these letters. In order not to add the remaining letters in bash, you need press the **Tab** key

Getting / changing personal information

To obtain personal information, should be used this commands:
who - shows who is currently logged into the system.
w - shows who is currently in the system and what he is doing.
whoami - Prints out the user's UID(the name of the user executing this command.)
id - prints extended user information (group, uid, gid).
finger - displays information about the user.

To change personal information:
chfn - change personal information displayed by finger.
chsh - change the commands interpratator .
passwd - change the user's password.

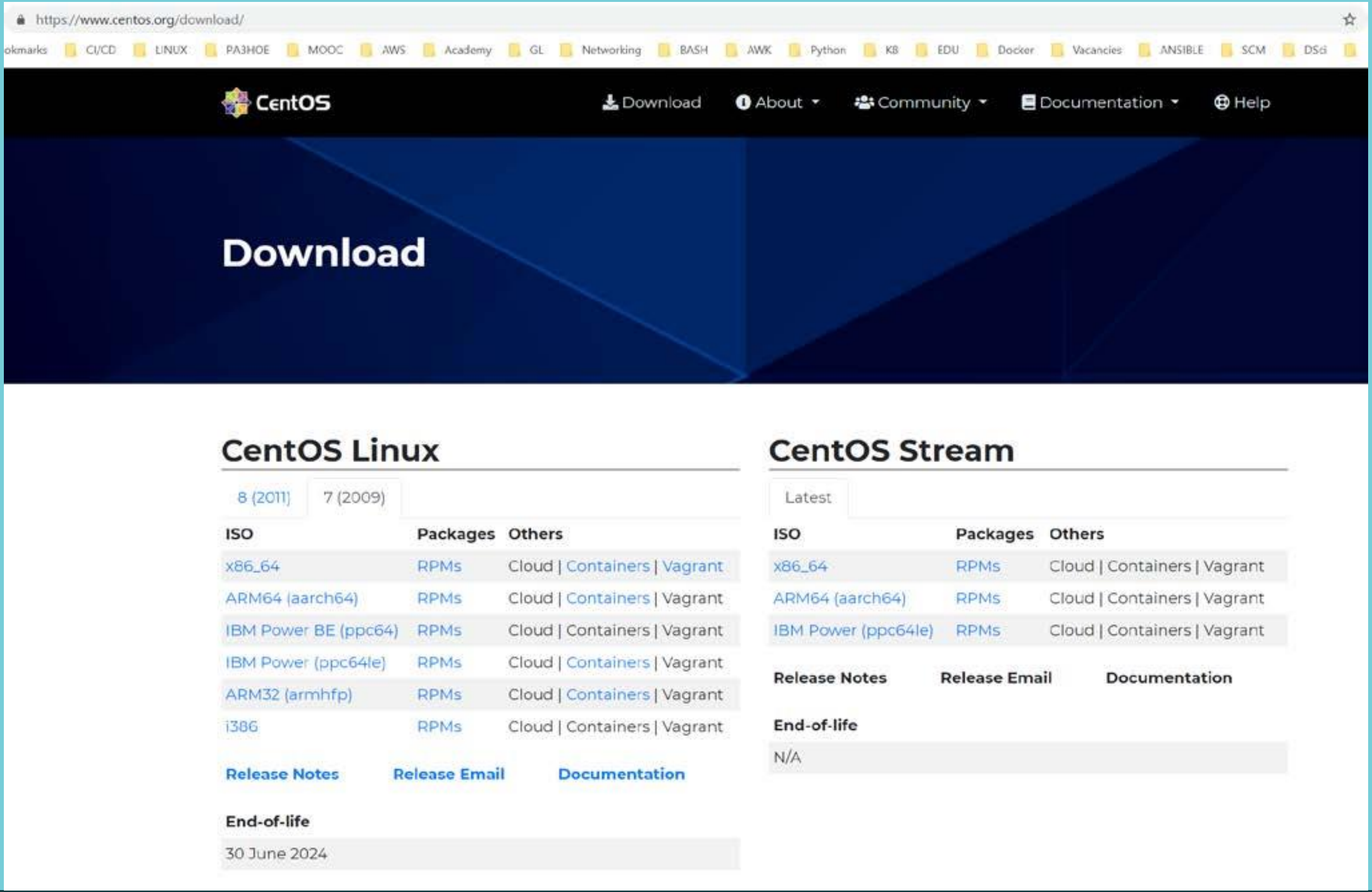
THE MAIN CHARACTERISTICS OF LINUX OSes

Reference guides are divided into sections - depending on the type of objects described. If the same term is described in more than one section, you must explicitly specify which one to use.

There are 8 sections in total:

1	Commands available to users
2	Unix and C system calls
3	C library routines for C programs
4	Special file names
5	File formats and conventions for files used by Unix
6	Games
7	Misc.
8	System administration commands and procedures

WHERE TO GET. https://www.centos.org/download/



Download

CentOS Linux

8 (2011) | 7 (2009)

ISO	Packages	Others
x86_64	RPMs	Cloud | Containers | Vagrant
ARM64 (aarch64)	RPMs	Cloud | Containers | Vagrant
IBM Power BE (ppc64)	RPMs	Cloud | Containers | Vagrant
IBM Power (ppc64le)	RPMs	Cloud | Containers | Vagrant
ARM32 (armhfp)	RPMs	Cloud | Containers | Vagrant
i386	RPMs	Cloud | Containers | Vagrant

Release Notes   Release Email   Documentation

End-of-life

30 June 2024

CentOS Stream

Latest

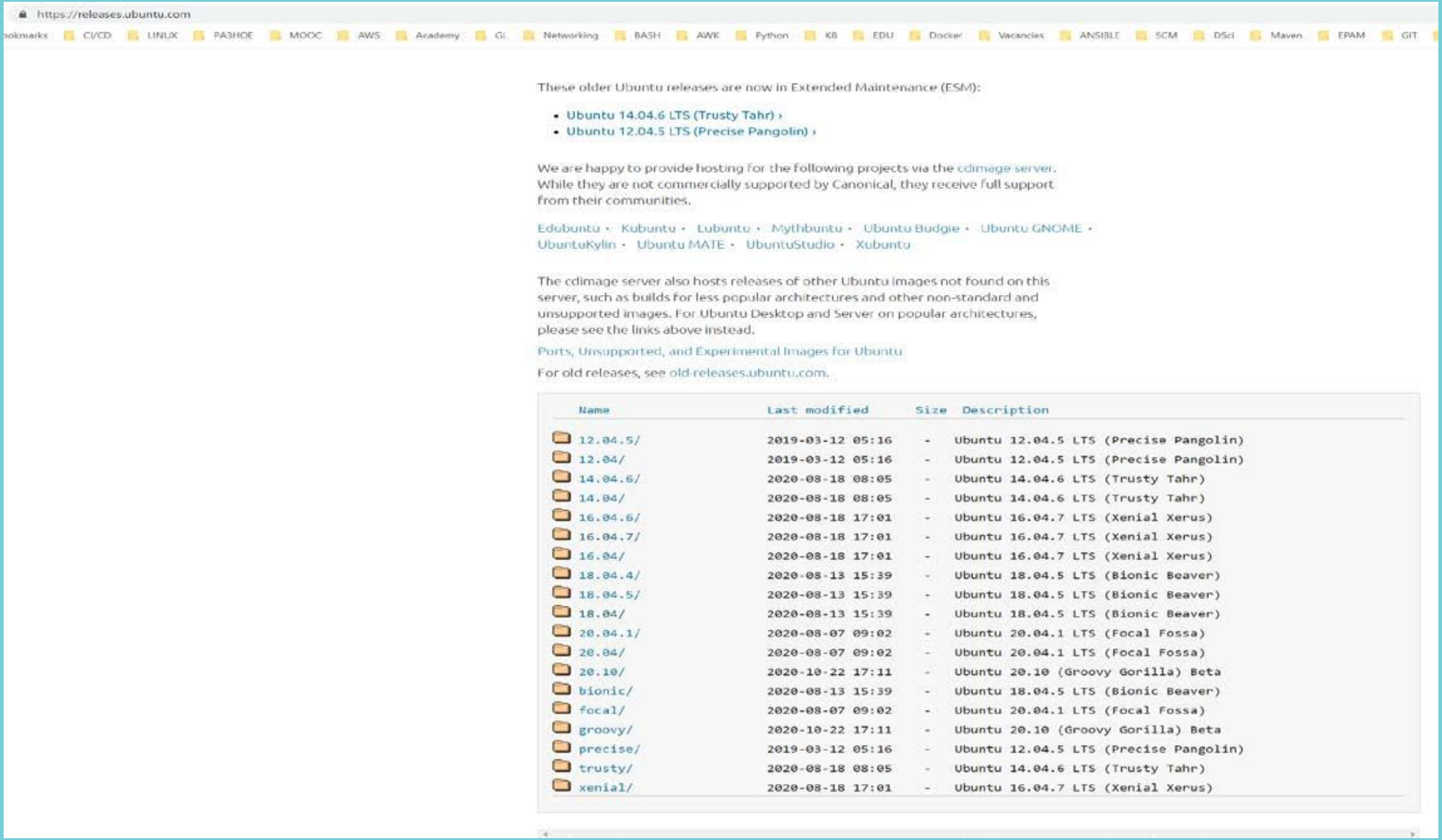
ISO	Packages	Others
x86_64	RPMs	Cloud | Containers | Vagrant
ARM64 (aarch64)	RPMs	Cloud | Containers | Vagrant
IBM Power (ppc64le)	RPMs	Cloud | Containers | Vagrant

Release Notes   Release Email   Documentation

End-of-life

N/A

WHERE TO GET. https://releases.ubuntu.com/


https://releases.ubuntu.com
okmarks CI/CD LINUX PA3HOE MOOC AWS Academy GL Networking BASH AWK Python KB EDU Docker Vacancies ANSIBLE SCM DSci Maven EPAM GIT

These older Ubuntu releases are now in Extended Maintenance (ESM):

- Ubuntu 14.04.6 LTS (Trusty Tahr) ›
- Ubuntu 12.04.5 LTS (Precise Pangolin) ›

We are happy to provide hosting for the following projects via the cdimage server. While they are not commercially supported by Canonical, they receive full support from their communities.

Edubuntu · Kubuntu · Lubuntu · Mythbuntu · Ubuntu Budgie · Ubuntu GNOME · UbuntuKylin · Ubuntu MATE · UbuntuStudio · Xubuntu

The cdimage server also hosts releases of other Ubuntu images not found on this server, such as builds for less popular architectures and other non-standard and unsupported images. For Ubuntu Desktop and Server on popular architectures, please see the links above instead.

Ports, Unsupported, and Experimental Images for Ubuntu

For old releases, see old-releases.ubuntu.com.

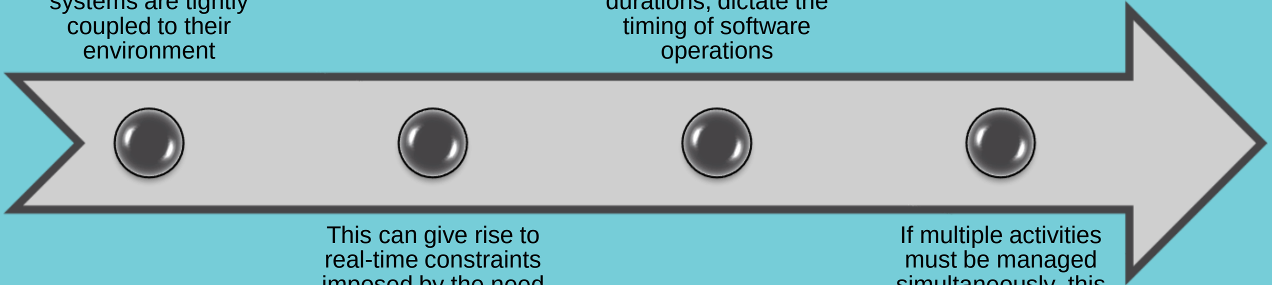
Name	Last modified	Size	Description
12.04.5/	2019-03-12 05:16	-	Ubuntu 12.04.5 LTS (Precise Pangolin)
12.04/	2019-03-12 05:16	-	Ubuntu 12.04.5 LTS (Precise Pangolin)
14.04.6/	2020-08-18 08:05	-	Ubuntu 14.04.6 LTS (Trusty Tahr)
14.04/	2020-08-18 08:05	-	Ubuntu 14.04.6 LTS (Trusty Tahr)
16.04.6/	2020-08-18 17:01	-	Ubuntu 16.04.7 LTS (Xenial Xerus)
16.04.7/	2020-08-18 17:01	-	Ubuntu 16.04.7 LTS (Xenial Xerus)
16.04/	2020-08-18 17:01	-	Ubuntu 16.04.7 LTS (Xenial Xerus)
18.04.4/	2020-08-13 15:39	-	Ubuntu 18.04.5 LTS (Bionic Beaver)
18.04.5/	2020-08-13 15:39	-	Ubuntu 18.04.5 LTS (Bionic Beaver)
18.04/	2020-08-13 15:39	-	Ubuntu 18.04.5 LTS (Bionic Beaver)
20.04.1/	2020-08-07 09:02	-	Ubuntu 20.04.1 LTS (Focal Fossa)
20.04/	2020-08-07 09:02	-	Ubuntu 20.04.1 LTS (Focal Fossa)
20.10/	2020-10-22 17:11	-	Ubuntu 20.10 (Groovy Gorilla) Beta
bionic/	2020-08-13 15:39	-	Ubuntu 18.04.5 LTS (Bionic Beaver)
focal/	2020-08-07 09:02	-	Ubuntu 20.04.1 LTS (Focal Fossa)
groovy/	2020-10-22 17:11	-	Ubuntu 20.10 (Groovy Gorilla) Beta
precise/	2019-03-12 05:16	-	Ubuntu 12.04.5 LTS (Precise Pangolin)
trusty/	2020-08-18 08:05	-	Ubuntu 14.04.6 LTS (Trusty Tahr)
xenial/	2020-08-18 17:01	-	Ubuntu 16.04.7 LTS (Xenial Xerus)

EMBEDDED SYSTEMS

Embedded System

- The term *embedded system* refers to the use of electronics and software within a product that has a specific function or set of functions, as opposed to a general-purpose computer
- An *embedded system* can also be defined as any device that includes a computer chip, but that is not a general-purpose workstation, or desktop or laptop computer
- Examples of types of devices with embedded systems include cell phones, digital cameras, video cameras, calculators, home security systems, washing machines, various automotive systems, tennis rackets, toothbrushes, and numerous types of sensors and actuators in automated systems

Embedded System



Often, embedded systems are tightly coupled to their environment

This can give rise to real-time constraints imposed by the need to interact with the environment

Constraints, such as required speeds of motion, required precision of measurement, and required time durations, dictate the timing of software operations

If multiple activities must be managed simultaneously, this imposes more complex real-time constraints

What makes a good Embedded OS ?

- Modular
- Configurable
- Scalable
- Wide CPU support
- Device Drivers
- Small size
- Etc…

Real Time in OS

- The ability of the operating system to provide a required level of service in a bounded response time.
  – POSIX standard 1003.1
- Hard & Soft Real Time.

What makes a good RTOS?

- Multi-threaded and pre-emptible
- Must support predictable thread synchronization mechanisms
- A system of priority inheritance must exist

Some Embedded OS flavors

- VxWorks
- pSOS
- QNX
- Integrity
- Palm OS
- Symbian OS

- Embedded Windows
- Pocket PC DOS
- Linux
- Android
- Cisco IOS & IOS-XR
- Apple iOS

Embedded Linux

- A version of Linux running in an embedded system
- Embedded devices typically require support for a specific set of devices, periphrals, and protocols, depending on the hardware that is present in a given device and the intended purpose of that device
- An embedded Linux distribution is a version of Linux to be customized for the size and hardware constraints of embedded devices
    - Includes software packages that support a variety of services and applications on those devices
    - An embedded Linux kernel will be far smaller than an ordinary Linux kernel

Commercial Embedded Linux & RTOS

- Raspbian
- Red Hat Embedded Linux
- FSMLabs - Open RT Linux
- LynuxWorks - BlueCat RT
- TimeSys - Linux/Real-Time
- uClinux
- Emdebian
- OpenWRT/LEDE

Open source Embedded Linux & RTOS

- ART Linux
- KURT
- Embedded Debian Project
- uCLinux – For CPU’s without MMU
- RTAI
- Etc.

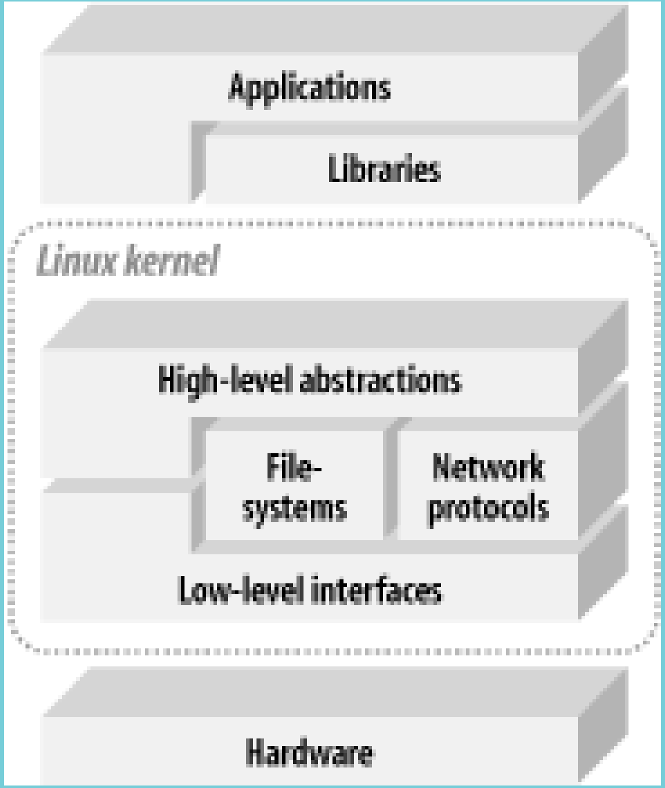
Special features of Linux

- Source code freely available
- Robust and reliable
- Modular, configurable, scalable
- Good support for Networking
- No runtime licenses
- Large pool of skilled developers
- Free software and tools

Core features of Linux

- CPU support (x86, ARM, PowerPC, MIPS,Etc.)
- Busses & Interfaces (ISA, PCI, PCMCA, VME, Parallel, SCSI, USB, IEEE1394, I2C)
- I/O (Keyboard, mouse, Display)
- Storage (ATA-ATAPI, DiskOnChip, CFI, RAM, ROM)
- File Systems (JFFS, FAT, EXT, NFS, Etc.)
- Networking (Ethernet, IrDa, 802.11x)

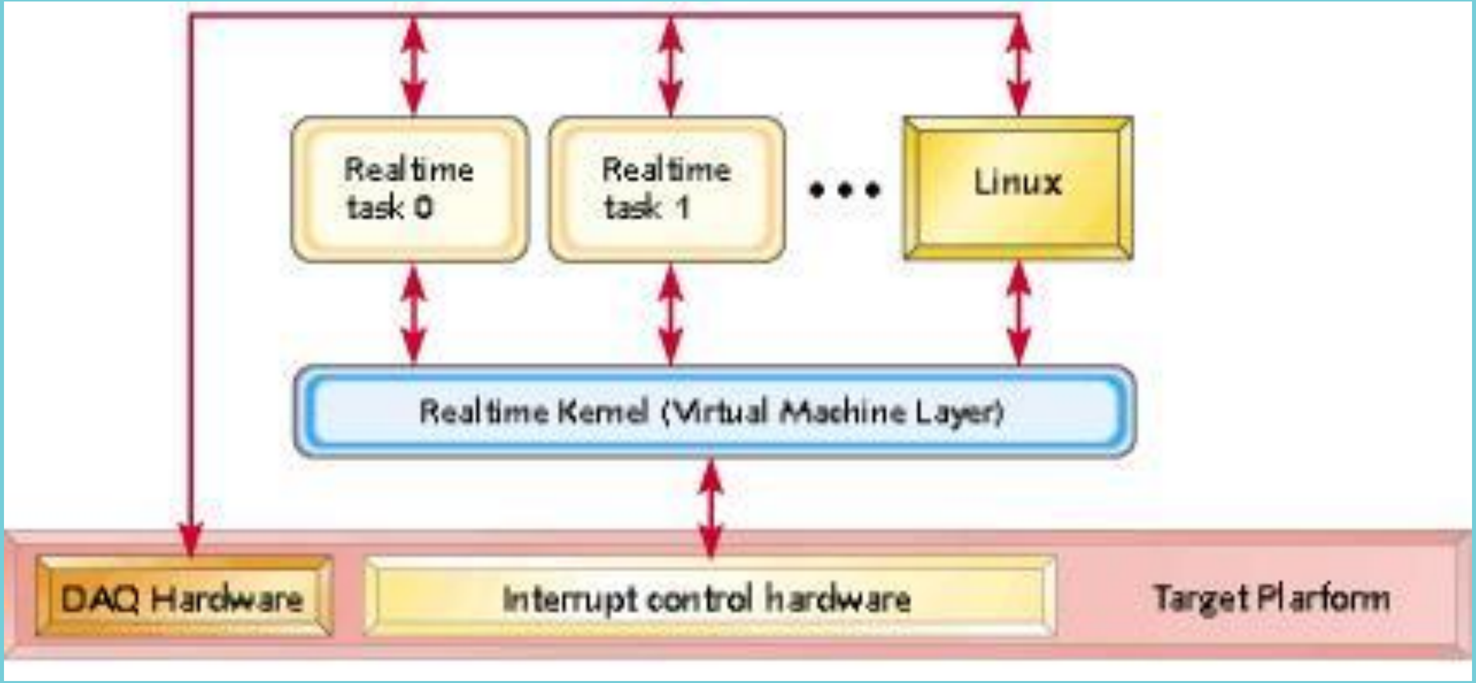
Architecture of Linux



RTLinux

- A “hard real-time” mini operating system
- runs Linux as it’s lowest priority execution thread
- Linux thread completely preemptible
- Real time threads and interrupt handlers never delayed by non-realtime operations
- Supports user level programming

RTLinux Architecture


Realtime task 0
Realtime task 1
Linux
Realtime Kernel (Virtual Machine Layer)
DAQ Hardware
Interrupt control hardware
Target Plarform

Development tools

- Compiler, assembler, linker, etc..
- Commercial
- Open Source.

Open Source Tool chain

- Kernel headers
- gcc – Compiler
- binutils – assembler,linker,debugger etc..
- glibc – Libraries
- Patches if any

Applications

- Industrial Controllers
- Mobiles, PDA, Media Centers.
- Telecomm and Networking Hardware
- Automobile Computers
- Robotics
- Vision Systems
- Etc.

QUESTIONS & ANSWERS

THANK YOU!