Integrarea sistemelor informatice

Suport curs nr. 10

Programator >> Arhitect

Integrarea soluțiilor software open source

2025-2026

Obiective

- Înțelegerea conceptului open source
- Analiza perspectivelor de integrare a soluțiilor open source
- Identificarea tipurilor de licențe open source

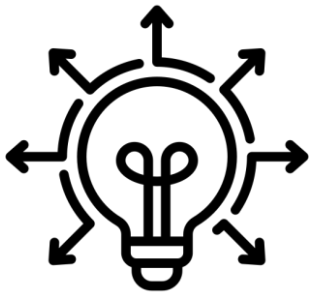
Note
Această secțiune de curs a fost dezvoltată în cadrul unui proiect susținut de 3 organizații americane:

USAID / CHF / Booz Allen Hamilton
Consultant: Kelly Looney

Actualizat 2023, 2025

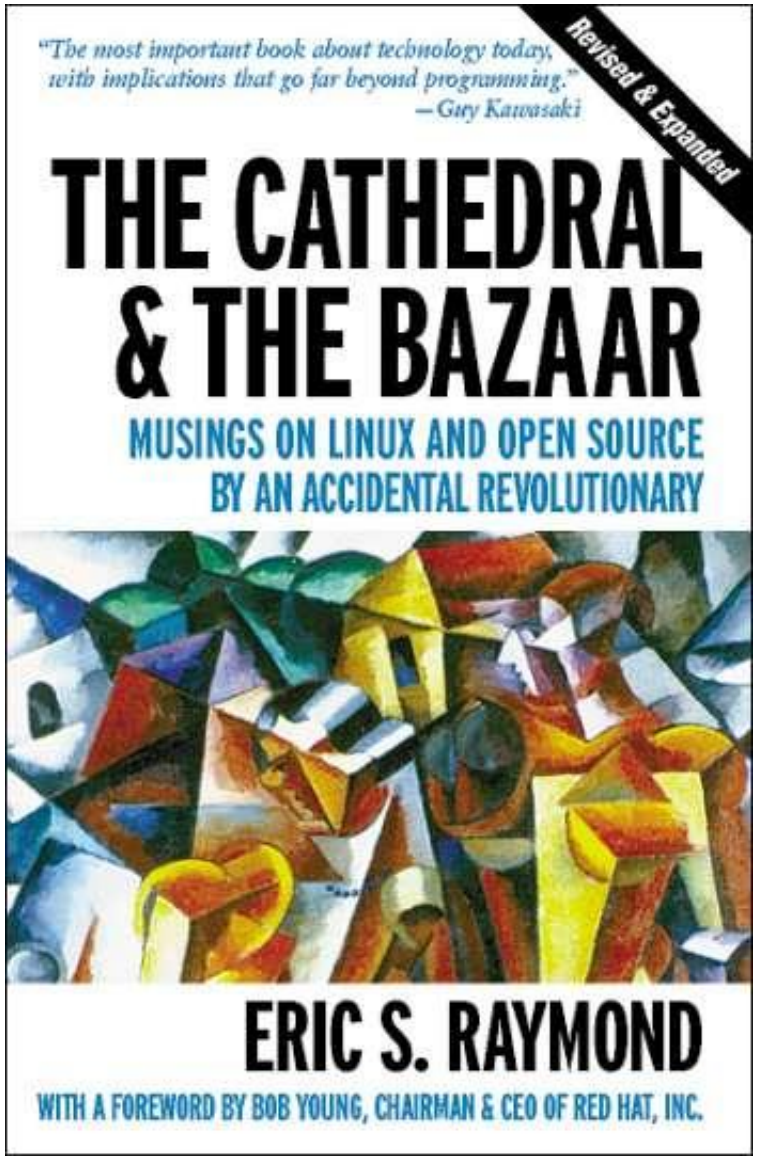
Open Source – Introduction

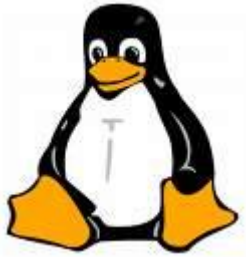
- When arbitrary programmers on the Internet can read, redistribute, and modify the source for a piece of software, it evolves.
- Open source advocates believe that the best way to reduce errors in software is to publish the source code on the Internet and allow anyone in the world to debug it.



History

- Origins of Open Source
  - UNIX and BSD
  - Free Software Foundation
- Linus Torvalds (Linux)
- The Cathedral & The Bazaar
  - Eric Raymond
- The Turning Point
  - Netscape, IBM, SUN, Oracle, …





Advantages

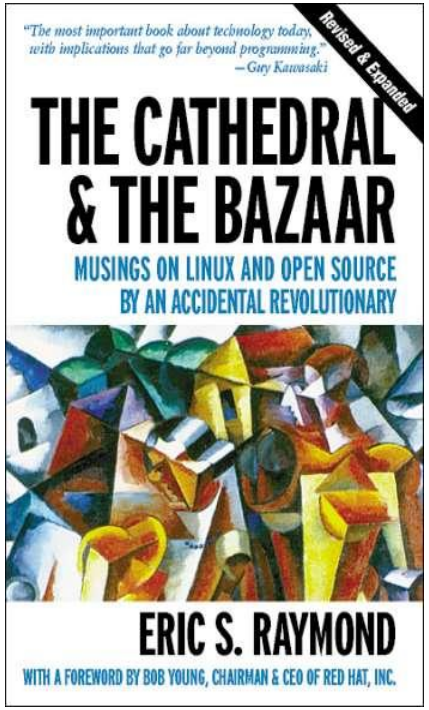
- Freely available source code
- Right to redistribute modifications and improvements
- No single entity decides the future of software
- High motivation for developers
- Product is released when it is ready

Disadvantages

- No guarantee on development goals
- Code contamination
- It is difficult to know if a project exists and its current status
- Many third-party software packages are not compatible with open source

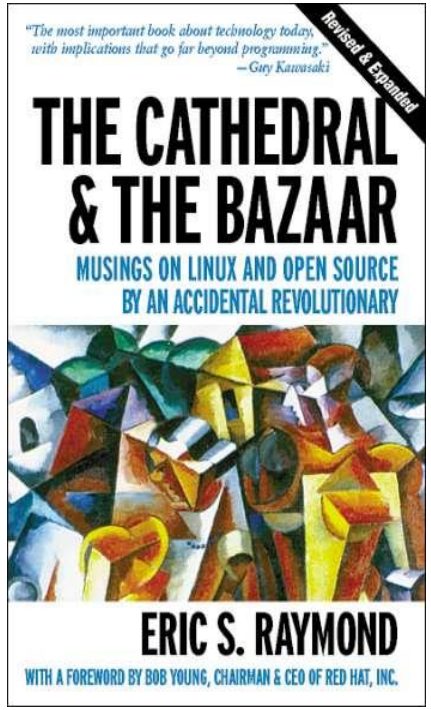
Open Source Models – The Cathedral

- Draws an analogy between traditional closed source development and a cathedral, in which there is a rigid hierarchy among developers, managers, testers, etc.
- E. Raymond (famous open source developer) originally believed that the most important software (operating systems and complex tools) needed to be built like cathedrals,
- carefully crafted by individual wizards or small bands of mages working in splendid isolation, with no beta to be released before its time.



Open Source Models – The Bazaar

- Open source projects similar to Middle Eastern bazaars, where numerous merchants hawk their wares loudly to passersby.
- Little hierarchy among contributors.
- Contributors compete to have their modifications inserted into the next release, bringing recognition and reputation.
- Describes Linus Torvalds' style of development as: release early and often, delegate everything you can, be open to the point of promiscuity.



Notes for the Bazaar Style (1)

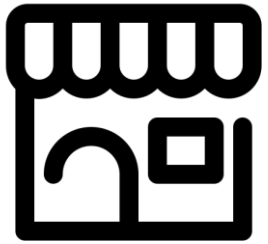
- One cannot code from the ground up in bazaar style.
- One can test, debug and improve in bazaar style, but it would be very hard to *originate* a project in bazaar mode.
- Your growing developer community needs to have something usable and testable to play with.

Notes for the Bazaar Style (2)

- To start community-building, what you need to be able to present is a *plausible promise*.
- Your program doesn't have to work particularly well. It can be crude, buggy, incomplete, and poorly documented.
- What it must not fail to do is (a) run, and (b) convince potential co-developers that it can be evolved into something really neat in the foreseeable future – e.g., through strong, attractive basic design.

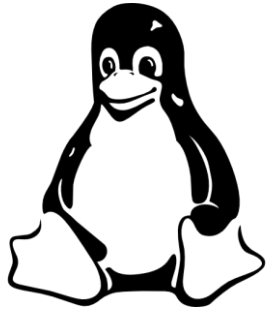
Notes for the Bazaar Style (3)

- It is not critical that the **coordinator** be able to originate designs of exceptional brilliance, but it is absolutely critical that the coordinator be able to *recognize good design ideas from others*.
- Must have good people and communications skills.
- In order to build a development community, you need to attract people, interest them in what you're doing, and keep them happy about the amount of work they're doing.



Internet and Open Source

- Linux was the first project to make a conscious and successful effort to use the entire *world* as its talent pool.
- The development of Linux coincided with the birth of the World Wide Web, and Linux was launched during the same period in 1993-1994 that saw the takeoff of the ISP industry and the explosion of mainstream interest in the Internet.



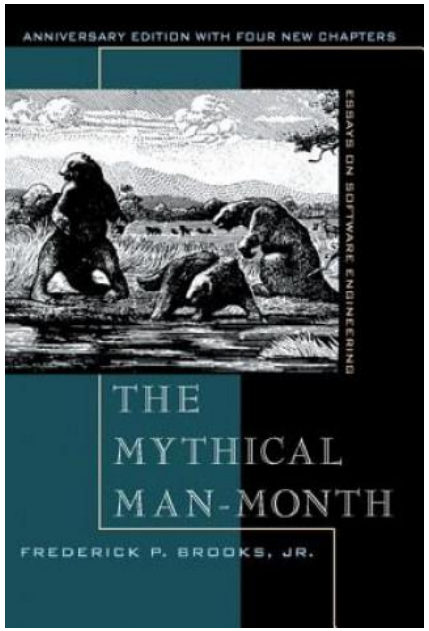
Open Source Timeline

GNU Manifesto	Free Software Foundation	Cygnus				The Cathedral and the Bazaar	Open source (term)	
1983	1985	1989	1991	1993	1995	1997	1998	1999
		GNU / GDB	Linux	Debian	Apache		Netscape	GNOME

http://eu.conecta.it/paper/Some_dates_open_source.html

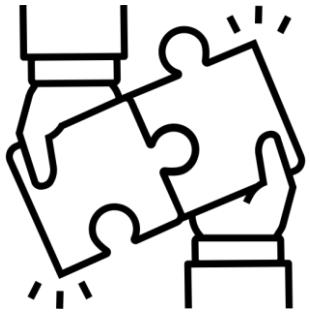
Open Source – Lessons (1)

- Good programmers know what to write. Great ones know what to rewrite (and reuse).
- "Plan to throw one away; you will, anyhow." (Fred Brooks, "The Mythical Man-Month", Chapter 11)
- This is and has been an impractical suggestion. Plan to build something as quickly as possible and modify it as needed in the future (not rewriting it).



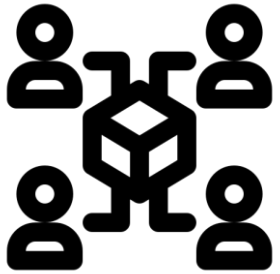
Open Source – Lessons (2)

- If you have the right attitude, interesting problems will find you.
- Treating your users as co-developers is your least-hassle route to rapid code improvement and effective debugging.
- When you lose interest in a project, your last duty to it is to hand it off to a competent successor.



Open Source – Lessons (3)

- Release early. Release often. And listen to your customers.
- Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly, and the fix be obvious to someone.
- Smart data structures and dumb code works a lot better than the other way around.

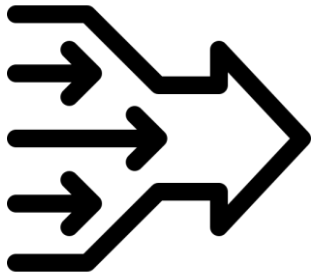


Open Source – Lessons (4)

- If you treat your beta-testers as if they're your most valuable resource, they will respond by becoming your most valuable resource.
- The next best thing to having good ideas is recognizing good ideas from your users. Sometimes the latter is better.
- Often, the most striking and innovative solutions come from realizing that your concept of the problem was wrong.

Open Source – Lessons (5)

- “Perfection (in design) is achieved not when there is nothing more to add, but rather when there is nothing more to take away.”
- Any tool should be useful in the expected way, but a truly great tool lends itself to uses you never expected.

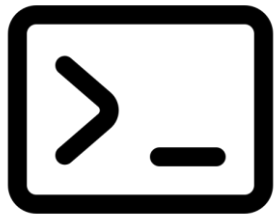


Brooks' Law

- Fred Brooks, in "The Mythical Man-Month" observed that programmer time is not fungible - adding developers to a late software project makes it later.
- He argued that the complexity and communication costs of a project rise with the square of the number of developers, while work done only rises linearly.
- This claim has since become known as "Brooks' Law" and is widely regarded as a truism.

Egoless Programming

- But if Brooks' Law were the whole picture, Linux would be impossible.
- Gerald Weinberg “The Psychology Of Computer Programming” supplied a vital correction to Brooks.
- His discussion of “egoless programming” observed that in shops where developers are not territorial about their code and encourage other people to look for bugs and potential improvements in it, improvement happens dramatically faster than elsewhere.



GNU – Free Software Foundation

- GNU - Free Software Foundation
  - Help your neighbor
  - Run the program for any purpose
  - Study how the program works
- vs. Commercial License
  - Restrictive License
  - vs. Shareware
  - No source code
- Richard Matthew Stallman, free software movement activist and programmer

Open Source Software

- Basic definition:
  - Software for which the source code is available to the public at no charge
  - In exchange, improvements made to the software by the public are expected to be submitted for potential inclusion in the software
- The use of the software is typically free, with commercial versions sometimes available
- Many open source software programs are protected by the **GNU General Public License**
  - see https://www.gnu.org/licenses/gpl-3.0.html

Open Source Software

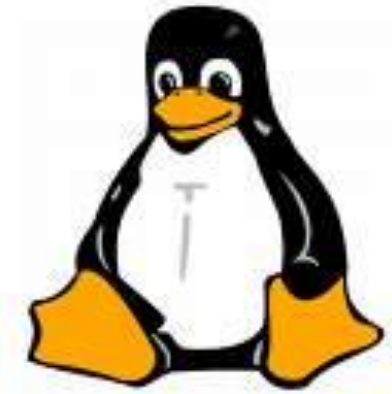
- Creation: Open source software is typically created by ad hoc groups of people, rather than by companies
  - A few products, like the MySQL database, are an exception
  - There are some very powerful groups that manage many of the most popular projects
    - The **Apache Software Foundation** (http://www.apache.org) is the dominant group

Open Source Software

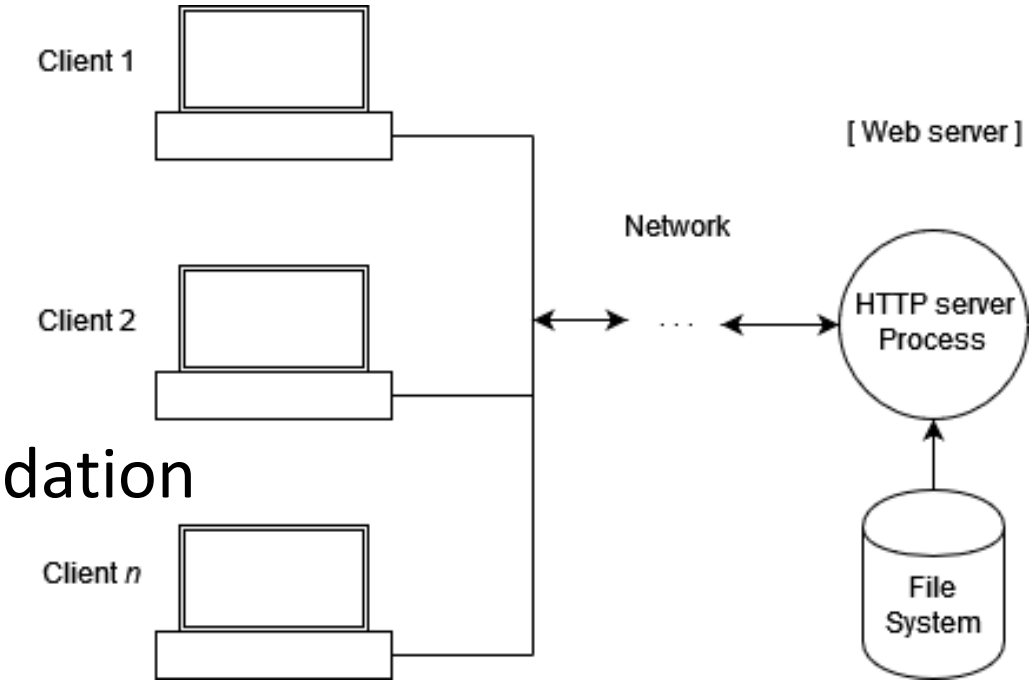
- Updates/Upgrades:
  - Updates/upgrades are **released by the group managing the project**
  - They often **incorporate code improvements from users**
- Support and Training:
  - These are provided by private companies
  - Often, participants in the groups creating and update the software work for these private companies

Some of the Biggest Open Source Projects

- Apache httpd
- MySQL
- PHP
- Linux



Apache httpd



- The world's leading Web server software
- Maintained by the Apache Software Foundation (http://www.apache.org)
- Online at http://httpd.apache.org
- Incredibly stable
- Without a doubt the most successful open source application
- Apache httpd is extraordinarily extensible – there are hundreds of modules available that extend httpd's functionality

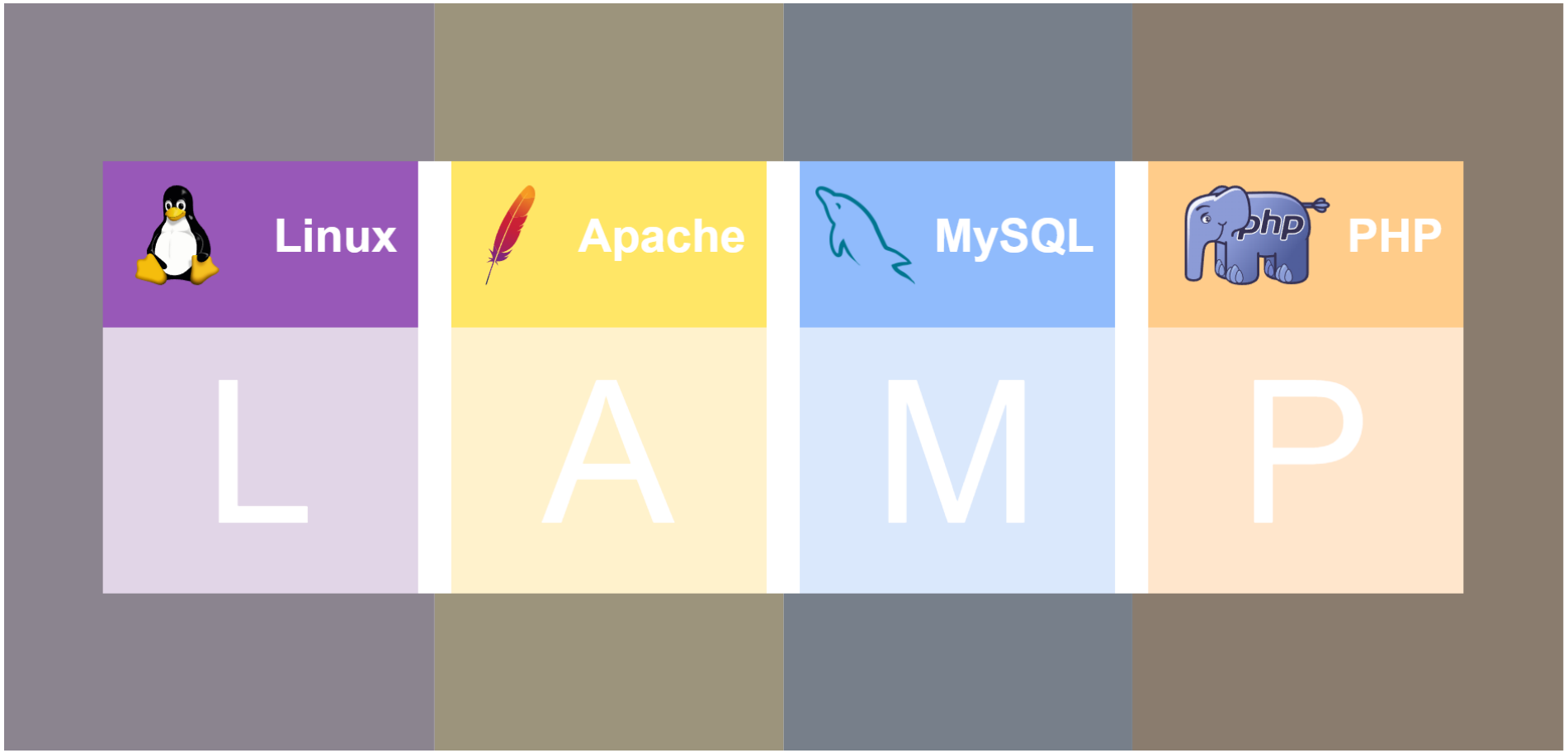
MySQL

- The world's fourth most popular database (after SQL Server, Oracle, and DB2)
- Maintained by Oracle
- Third parties provider GUIs for interacting with MySQL
  - A solid, free, Web-based one is phpMyAdmin, found at http://www.phpmyadmin.net
- For Web development, the MySQL database is often used in conjunction with the PHP scripting language

PHP

- Short for “Hypertext Preprocessor”
  - Found online at http://www.php.net
- PHP is a language used primarily for writing applications that run on Web servers
  - PHP serves to integrate the Web site with databases and other back-end applications (such as email servers)
- PHP has a good reputation for being fast and stable.
- Before modern web architectures and frameworks, it has been used in many current and legacy projects.

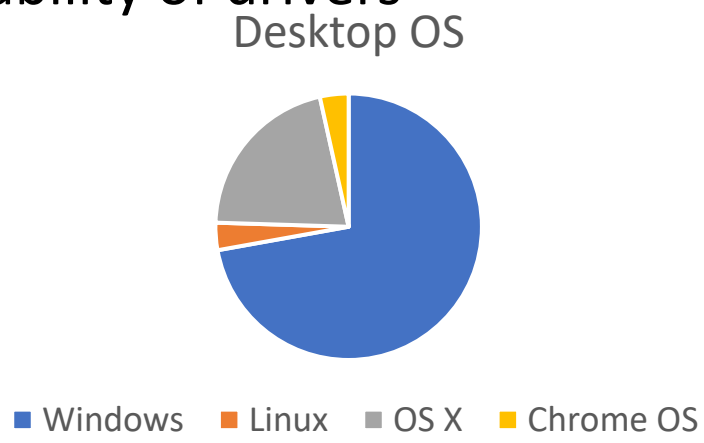
LAMP

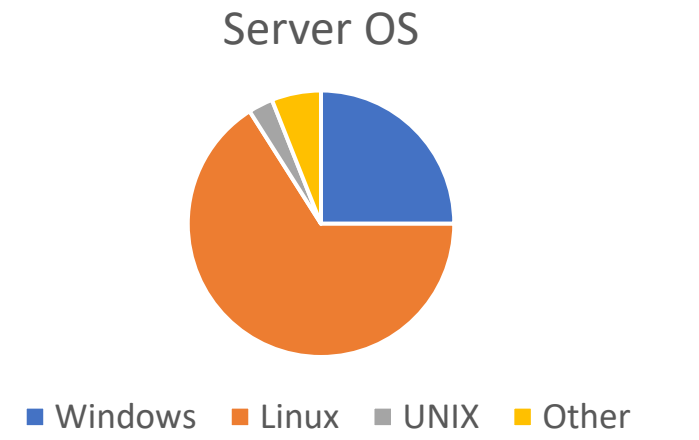


What is Lamp Stack? A Brief Introduction, emizentech, 2022

Linux

- Linux has experienced exponential growth as a platform for servers, especially Web servers
- Adoption of Linux on the desktop has been vastly slower due to:
  - Ease of use (increasingly overcome by shells such as GNOME and KDE)
  - Availability of applications (OpenOffice is great but is hardly MS Office)
  - Availability of drivers





SourceForge

- The world's largest site for the management and release of open source projects
- Online at http://www.sourceforge.net
- Typically the home of smaller (but nonetheless very significant) open source projects
  - phpMyAdmin is hosted here, for example

Hot Topics of Debate

- The Software is Free, but…
- More secure?
- Is the total cost of ownership lower?
- Viable in the long-term?

The Software is Free, but…

- Numerous companies often create add-ons that are sold commercially (such as the various front-ends for MySQL)
- Other commercial markets built around open source software
  - Support
  - Training
  - Documentation
  - Consulting and Development
- In some cases, open source software is not as easy to use as its commercial cousins, creating opportunities in the above areas

More secure?

Pros

- Source code is open to inspection and feedback by potentially a greater number of people
- Fixes can come from a wide variety of sources

Cons

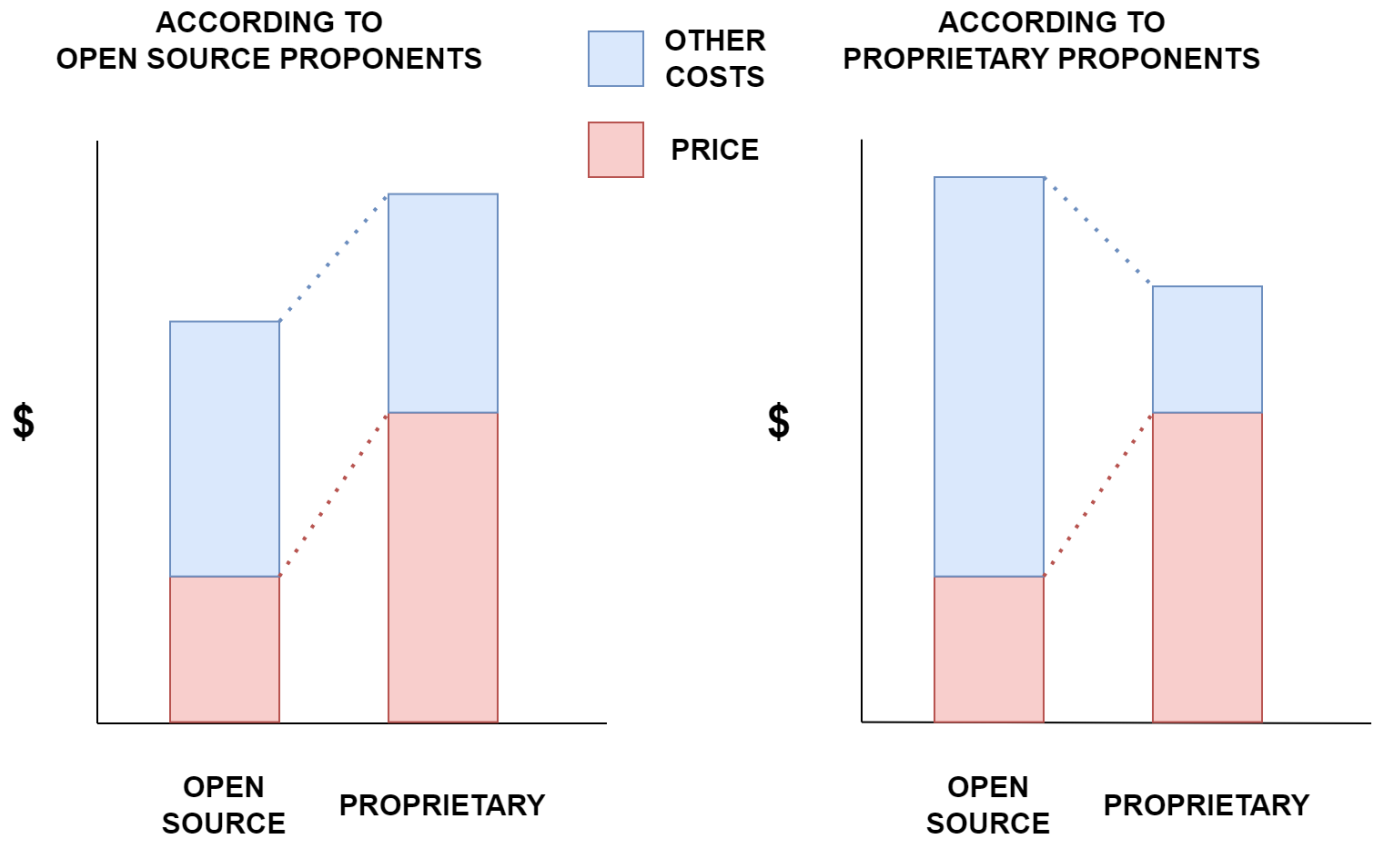
- Some open source projects have informal QA processes
- Developers aren’t paid to plug security holes
- Some people feel that open source lays bare any flaws

Is the total cost of ownership lower?

Open source proponents and proprietary companies disagree on the total cost of ownership.

- Proponents claim that even if open source requires more expertise, **the total cost (including deployment) is ultimately lower**
- Proprietary companies argue that the upfront cost is certainly lower, but since many of these projects are more complicated than their commercial counterparts, **the total cost (including training and expertise) is ultimately higher**

Differing Views of Total Cost of Ownership



Reproduced from “Open Options” developed by the Northwest Regional Educational Library, Portland, Oregon

Viable in the long-term?

- The biggest projects are 100% viable
  - Linux, Apache, MySQL, PHP, Tomcat, and the other largest projects have huge installed bases and aren’t going anywhere soon
- Your main risk is with smaller projects that may be superseded by competitors or on which development may go idle
  - This risk exists in the commercial world, too

Major Players: Business

- **Red Hat** – pioneer in making money out of free software – enterprise-grade open source solutions, now acquired by IBM
- **IBM** – earliest in the major firms to embrace open-source software (Linux kernel contributor, Apache Foundation projects)
- **Oracle** – supports open source products (MySQL, OpenJDK)
- **Microsoft** – acquired GitHub, provides open source technologies (.NET Core, TypeScript, Visual Studio Code)
- **Facebook** – open sourced key technologies (React, PyTorch, GraphQL)
- **Google** – major contributor to open source community (Android, Kubernetes, Chromium)

Major Players: Government

- **ROSEdu (Romanian Open Source Education)** – platform promotes the use of open source software in education
- **Code.gov (USA)** – platform promotes and facilitates the use of open source software within federal agencies
- **Open Government Data Platform (India)** – open data portal that provides access to a wide range of government resources
- **Brazil's Public Software Portal** – integrated platform for collaborative development based on open source software developed or used by government agencies

Business Model

- Optimization Strategy (premium personalized/optimized solutions)
- Dual License Strategy (open source and proprietary alternative)
- Consulting Strategy (paid consulting, training, and technical support)
- Subscription Strategy (access to premium features)
- Patronage Strategy (crowdfunding)
- Hosted Strategy (hosting services for infrastructure management)
- Embedded Strategy (integrating with commercial products)

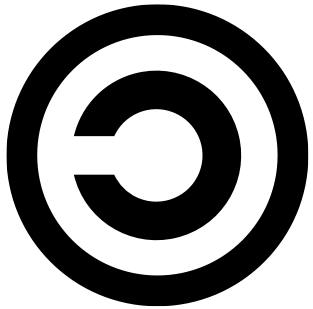
Licensing

- Distribution of License
  - License Must Not Be Specific to a Product
  - License Must Not Restrict Other Software
  - License Must Be Technology Neutral
- Free Software

  GNU General Public License
- Open Source

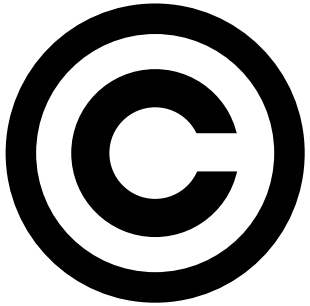
  GPL, LGPL, BSD, MIT, Mozilla
- Proprietary Software

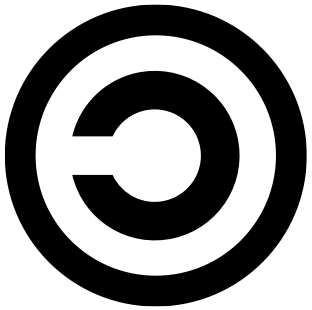
  End User License Agreement (EULA)



Copyright vs Copyleft

- A **copyright** is the exclusive legal right of a creator to reproduce, prepare derivative works, distribute, perform, display, sell, lend, or rent their creations.
- **Copyleft** is a general method for making a program or other work free and requiring all modified and extended versions of the program to be free as well.





Open Source Legal Issues

- Defining Terms
  - Software as Intellectual Property
  - Proprietary Software
  - Open Source Software
  - Freeware
- Advantages and Disadvantages
- Sharing Software Examples
- Best Value Procurement

Software as IP

- Software is valuable intellectual property
- A software license is the contract between the software owner and the licensee defining terms of use of software
- Software owners also have enumerated rights under the law to control the use and distribution of their property (with few exceptions)
- The Digital Millennium Copyright Act (DMCA) and contract law typically protect software owners' rights

Proprietary Software License

- Restrictions on dissemination. Licensee and users strictly defined. Licensee has no right to share with those not defined as licensee users in license
- Licensor indemnifies licensees against third party infringement claims
- Often, have to sign a new license each time new licensee obtains the code

Open Source Software License

- Generally, software provided “AS-IS” with no warranties, warranties excluded
- No indemnification (compensation for harm or loss)
- No maintenance or support

GNU “General Public License” (GPL)

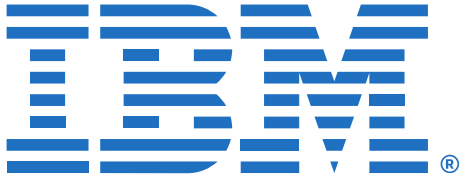
- No standard open source license, but GPL most widely used (roughly 85% of open source software)
- Terms include:
  - User freedom to distribute and/or modify
  - Requirement that original and modified source code be **always made available** to the world **under the terms of the original license**
  - **Must retain copyright notices and warranty disclaimers**
  - Does not include grant of patent licenses
  - Extremely **viral license**

Free as in Freedom

Examples: The Mozilla Public License (MPL)

- Developed by Netscape (now Mozilla Foundation) for the Mozilla browser
- Terms include:
  - Very similar to the GPL but,
  - **Can charge royalties for modified versions**
  - Can include source code within larger works licensed under different license types, thus license does not transfer to all downstream projects
  - **Must retain copyright notices and warranty disclaimers**
  - May provide additional warranties to downstream users but may have to indemnify original developer for any claims arising as a result
  - **Includes grant patent licenses**
  - **Less viral** than the GPL

Examples: The IBM Public License



- Terms include:
  - User freedom to distribute and/or modify
  - **No requirement for source code availability** in downstream distribution (shared/adapted/modified software)
  - **The program can be distributed in executable form thus allowing** downstream users to develop, sell, and install customized software packages without having to make all customizations available to the world
  - **Must retain all copyright notices and warranty disclaimers**
  - **Includes grant of patent licenses**

Example: The Apache Software License

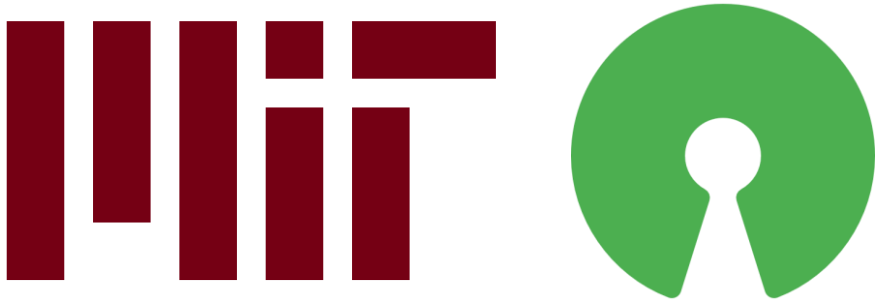
- Governs the Apache web-server software.
- Apache Software Foundation
- Terms include:
  - User freedom to distribute and/or modify
  - **No requirement for source code to be made available** to the world in downstream distribution
  - **Must retain all copyright notices and warranty disclaimers**
  - **Not a viral license**

Example: The FreeBSD License

- Unrestrictive license:
  - Only requires preservation of copyright notices and warranty disclaimers.
- Examples: BSD, ISC, MIT License

FreeBSD

Example: The MIT License



Copyright (c) <year> <copyright holders>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Freeware

- Freeware is free software, software that the licensee can use without paying a license fee
- Free software may be **proprietary** software for which source code is not provided (Adobe Acrobat) or **open source** software (Linux)
- **Note: Not all open source software is free / not all proprietary software is licensed for a fee**

Advantages for Proprietary Software

- **Indemnification** (compensation for harm or loss)
- **Maintenance and support** (by agreement / contract)
- Licensee doesn’t need to have open source savvy **staff**
- **Licensees’ rights** / protection applies:
  - media is defective
  - software contains viruses, backdoors, etc.
  - product fails to meet written technical / business specifications

Disadvantages for Proprietary Software

- **Cost** (license fee, product bundling, e.g., Microsoft Office)
- Licensee cannot **modify** or enhance the code
- Often not built to open standards, leading to **interoperability problems**
- **Closed** for development and information sharing in open source communities
- Some proprietary code is not as good as its open source counterparts

Advantages for Open Source Software

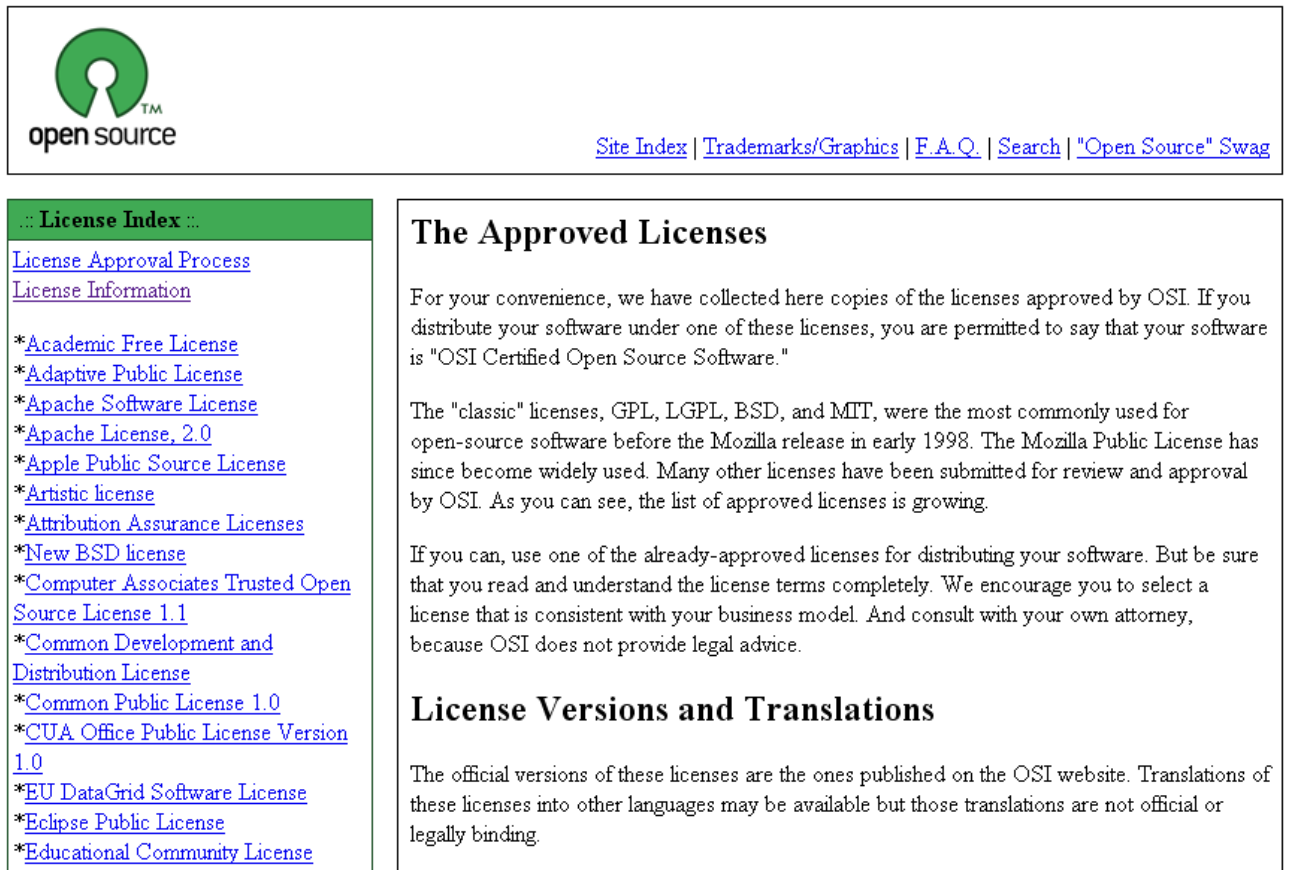
- **Price**: Free or low license fees
- **Availability** of source code coupled with permission to make modifications
- Access open source **development community –** continuing improvement / development
- More likely to be built to **open standards** / interoperable with other open source systems

Disadvantages of Open Source Software

- **No indemnification**; if a third party claims that licensee is using code that the third party developed, the licensee has no one to pay his legal fees and damage award (*SCO v. IBM)*
- **No maintenance and support** (unless purchased separately)
- **No warranties** regarding media, viruses, and performance
- **Trained staff** must be open source savvy
- **Note:** Many disadvantages of open source licensing can be addressed by procuring open source software through a third party vendor that will provide maintenance, support, additional warranties, etc.

Online resources

- For full open source license texts see www.opensource.org/licenses/index.php



Resources

- http://www.opensource.org/docs/definition.php
- http://www.netc.org/openoptions/pros_cons/comparing.html
- http://www.nber.org/papers/w7600
- http://www.gnu.org/copyleft/gpl.html
- http://danny.oz.au/free-software/advocacy/against_IP.html
- http://blogs.zdnet.com/BTL/?p=2019&part=rss&tag=feed&subj=zdblog
- http://www.itconversations.com/shows/detail657.html