Introductory Lecture & FLOSS

Lecture 1

Introduction

Organization

- Lectures:
  - Weekly lecture to cover course materials (until Christmas)
  - Lectures take place on Tuesdays, 17:00–19:00, Argentinierstraße 8, Seminarraum/Bibliothek 194-05
  - Attendance is mandatory
- Group project:
  - In groups of 4 students
  - 3 meetings with lecturers during the semester (week 44/2019, week 48/2019, week 2/2020)
  - Final presentations at the end of January (week 4/2020)
- Final paper:
  - In groups of 2 students
  - Final presentations at the end of January (week 5/2020)
  - Deadline: **Sunday, February 9, 2020, 23:59 CET** (no exceptions!)

Organization

- Grading:
  - 50% group project
  - 35% seminar paper
  - 15% participation during lectures
  - All course components need to be passed in order to pass the overall course!
- Course materials:
  - Will be provided at https://free-and-open-technologies.github.io
- For further questions:
  - Email christoph.derndorfer@tuwien.ac.at and lukas.f.lang@tuwien.ac.at

Lecture outline

1. FLOSS (Free/Libre and Open Source Software)
2. Open Hardware
3. Open Data
4. Open Content/Open Educational Resources
5. Open Science/Research
6. Open Access
7. Open Spaces/Open Practices: Metalab Vienna
8. Guest Lecture: Stefanie Wuschitz (Mz* Baltazar's Lab)

Group project

- Goal:
  - Extend, contribute to, or create a new open project within scope of lecture topics
- Choose topic from a list (see course website) or (even better) suggest your own:
  - Groups of 4 students
  - Send a 1-page proposal until **Friday, October 25**, via email to **both** lecturers
    - Define the idea, goal, (potential) impact, requirements, and estimated effort
    - State deliverables (should be broken down into three milestones to discuss in meetings)
- Requirements:
  - Open and accessible (Git repository, openly licensed) → others can access/use/study/extend
  - Use time sheet to track and compare estimated vs. actual time spent → you/others can learn
  - Transparent process (time sheets, Git commits) → others can re-create
  - Document your project (code comments, tutorials, videos, etc.) → others can understand

Group project

- 3 group meetings (45 minutes) with lecturers:
  - Week 44/2019: discussion of your project idea
  - Week 48/2019: progress review and feedback, first prototype/draft/results
  - Week 02/2020: review and feedback for project finalisation, ideas for presentation/demo
- Final presentations and demos at the end of January
  - Week 04/2020: max. 20 minutes & 5 minutes Q&A → 4 hours total
- Example projects
  - Analyze contributors and funders of free/libre and open-source software
  - Organizing a Wikithon
  - Low-cost open-source microscopy
  - Develop a new application based on open data
  - See https://free-and-open-technologies.github.io for a full list

Final paper

- Seminar paper:
  - Study literature on a lecture topic of your choice
  - Demonstrate understanding (and can be related to your group project)
  - Deadline: **Sunday, February 9, 2020, 23:59 CET** (no exceptions!)
  - Length: 10-12 pages (max. 1 page references)
  - Scientific standard: citations, quotes, etc. != project description
  - Source suggestions will be provided for each lecture topic
  - LaTeX template will be provided
  - Writing process and tool must be transparent and openly documented (e.g. Bitbucket or GitHub)
  - **Goal: Make all papers available on the course website under Creative Commons license**
- Presentation:
  - At the end of January (week 5/2020)
  - Length: max. 15 minutes & 5 minutes Q&A
  - Mandatory attendance

FLOSS

Free/Libre and Open Source Software

Open source software in numbers

- 2019: IBM acquires Red Hat for $34 billion
- 2019: MongoDB market capitalization: > $6 billion
- 2018: Microsoft acquires GitHub for $7.5 billion
- 2008: Oracle acquires MySQL AB for $1 billion

Famous open source projects

- CMS: Joomla, Wordpress
- Databases: MariaDB/MySQL, MongoDB
- DevOps: Ansible, Docker, Puppet
- Operating systems: GNU, Linux kernel, Android Open Source Project, Red Hat Enterprise Linux
- Productivity: Firefox, LibreOffice/OpenOffice, Thunderbird
- Servers: Apache Tomcat, Nginx, ownCloud
- Programming: Python, Scala, Ruby
- Others: Git, Hadoop, OpenCV, TensorFlow, etc.

[1] Wikipedia, “List of free and open-source software packages“, accessed 2019/10/07

Definitions: What is open source software?

No universally agreed-upon definition of open source software

Versatility: definitions may change over time

Two main movements:

- Free software movement – driven by the Free Software Foundation (FSF)
- Open source software movement – driven by Open Source Initiative (OSI)

The Free Software Definition:

- Idealistic point of view (Motto: “Free Software, Free Society”)
- Defines four essential freedoms
- Followed by the GNU project

The Open Source Definition:

- Pragmatic point of view
- Defines 10 criteria for open source software
- Followed by e.g. Debian, Linux Foundation, Mozilla Foundation, Wikimedia Foundation

[1] Open Source Initiative, “The Open Source Definition”, 2007/03/22

[2] Free Software Foundation, “What is free software?”, V. 1.165, 2019/07/30

The Free Software Definition

Richard Stallman developed GNU OS as the first free/open source software in 1984 and subsequently defined four essential freedoms [1]:

- You have the freedom **to run** the program, for any purpose.
- You have the freedom **to modify** the program to suit your needs.
  - To make this freedom effective in practice, you must have access to the source code, since making changes in a program without having the source code is exceedingly difficult.
- You have the freedom **to redistribute copies**, either gratis or for a fee.
- You have the freedom **to distribute modified versions** of the program, so that the community can benefit from your improvements.

[1] Richard Stallman, “The GNU Operating System and the Free Software Movement,” in Open Sources: Voices From the Open Source Revolution, edited by Chris DiBona, Sam Ockman, and Mark Stone (Sebastopol, CA: O’Reilly & Associates, 1999), p. 56.

The Open Source Definition

The OSI was founded by Bruce Perens and Eric S. Raymond in 1998 as a branch of the free-software movement.

Ten criteria for open source software [1]:

- Free redistribution
- Source code
- Derived works
- Integrity of the author's source code
- No discrimination against persons or groups
- No discrimination against fields of endeavor
- Distribution of license
- License must not be specific to a product
- License must not restrict other software
- License must be technology-neutral

[1] Open Source Initiative, "The Open Source Definition", 2007/03/22

Main differences: open source vs. free software

Same values with mainly philosophical differences:

- English term “free” is ambiguous: “free software” is a matter of liberty, not price
- Famous quote by Richard Stallman: “Think free as in free speech, not free beer.”
- Use of the French word “libre” (i.e. "with little or no restriction") makes this more explicit [1]
- Both demand that software must be available for commercial use, development, distribution

Accept same software licenses with minor exceptions:

- OSI accepts some licenses that FSF considers too restrictive (see table of approvals in [2])
- Example: NASA Open Source Agreement 1.x requires changes in code to be of “original creation”

Stallman considers open source as superset of free software:

- Example: Development of DRM software as open source, despite putting restrictions on users [3]

Neutral terms:

- FLOSS: Free/Libre and Open Source Software ← preferred term in this course
- FOSS: Free and Open Source Software

[1] Richard Stallman, “FLOSS and FOSS”, 2016/11/18
[2] Wikipedia, “Comparison of free and open-source software licenses”, accessed 2019/10/03
[3] Richard Stallman, “Why Open Source misses the point of Free Software”, 2019/04/28

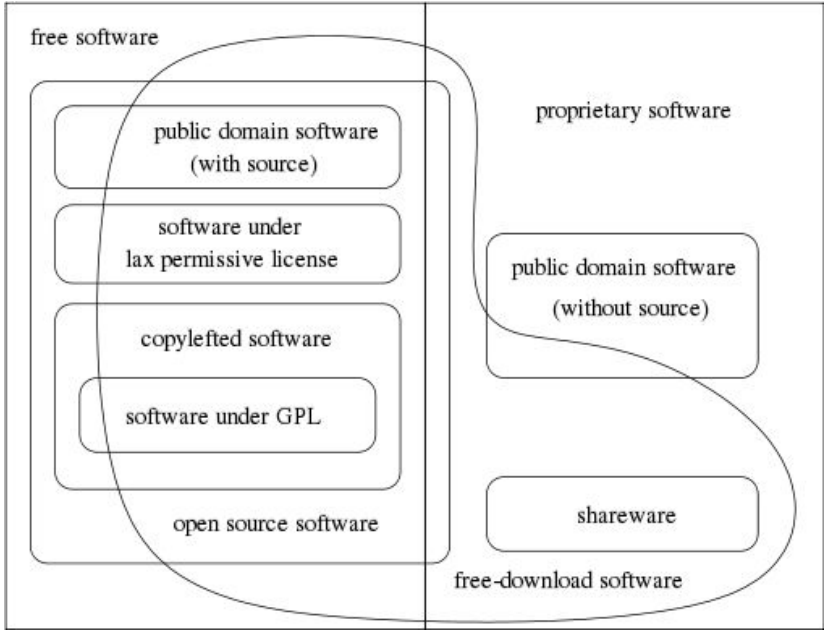
Main differences: open source vs. free software

Practical differences [1]:

- Almost all existing free software source code would qualify as open source
- Nearly all open source software is free software:
  - Exception: “Open Watcom” (C/C++ IDE) is open source but nonfree
  - License does not allow making a modified version and using it privately e.g. deploy it
- “Weak” licenses allow restrictions on executables:
  - Example: MS Visual Studio is open source but not free (“gratis”) software
  - But: users can compile, use, and distribute
- “Tivoization”:
  - Devices (“tyrants”) block users from installing self-compiled executables by checking signatures, effectively making free software nonfree
  - Example: TiVo digital video recorder, Android contains tivoized executables of Linux (allowed under GNU GPL v2, but is prohibited under v3)

[1] Richard Stallman, “Why Open Source misses the point of Free Software”, 2019/04/28

What is free/libre and open source software?



Encircled: “gratis” software (freeware; available free of charge)

[1] Image by Chao-Kuei, licensed under CC BY-SA 2.0
[2] Free Software Foundation, “Categories of free and non-free software”, 2019/02/21

Key figures

Richard Matthew Stallman (RMS)

- Creator of free software via GNU OS (1984), Free Software Foundation (FSF) (1985), GPL license (1989), etc.
- Recently resigned from MIT and FSF after controversial comments around Epstein

Linus Torvalds

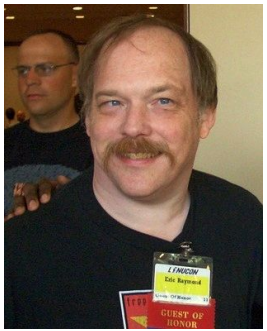
- Original creator of the Linux kernel (1991) as a student at University of Helsinki
- Continues to be the kernel's maintainer

[1] By Preliminares 2013, CC BY-SA 2.0, https://commons.wikimedia.org/w/index.php?curid=23722768
[2] By Krd (photo)Von Sprat (crop/extraction) - File:LinuxCon Europe Linus Torvalds 03.jpg, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=54706023

Key figures

Eric S. Raymond

- Author of “The Cathedral and the Bazaar” [3]
- The 1997 essay is attributed to convincing Netscape to release its browser source code in 1998
- Co-founder of the Open Source Initiative (OSI) in 1998



Christine Peterson

- Coined the term “open source” in 1998 [4]
- Co-Founder, Foresight Institute

[1] By jerone2 - cimg0230, CC BY-SA 2.0, https://commons.wikimedia.org/w/index.php?curid=3453961
[2] By Randy Stewart - https://www.flickr.com/photos/stewtopia/3843277009/, CC BY-SA 2.0, https://commons.wikimedia.org/w/index.php?curid=74490182
[3] Eric Steven Raymond, The Cathedral and the Bazaar, accessed 2019/10/06
[4] Open Source Initiative, History of the OSI, accessed 2019/10/06

History of FLOSS

1950s/1960s: Openness and co-operation:

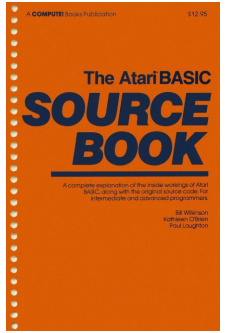
- Software was not widely considered as a commodity
- Mainly practiced by academics and corporate researchers
- Source code often shipped/distributed with machine code (allowed bug fixes and adaptations)
    - Examples: A-2 compiler/linker tool for UNIVAC, IBM mainframe software

1970s/1980s: Software becomes an asset:

- Growing software industry (OS, compilers) in late 1960s and early 1970s
- 1974: US Commission on New Technological Uses of Copyrighted Works (CONTU) decided that computer programs are subject to copyright
- Selling of software licenses started (through copyrights, trademarks, leasing contracts)
- 1983: IBM stopped distributing source code with software products
- Trend to sell executables
- TeX is an example of free software that is still widely used

[1] Grace Murray Hopper, in her office in Washington DC, 1978, by Lynn Gilbert, CC BY-SA 4.0
[2] Donald Knuth, 2005, by Jacob Appelbaum, CC BY-SA 2.5
[3] Wikipedia, “History of free and open-source software”, accessed 2019/10/03

History of FLOSS



1980s: Source code mainly shared by hackers and hobbyists:

- Magazines and books (e.g. The Atari BASIC source book [1])
- Online communities (e.g. USENET) for sharing and modifying source code started forming

1983: Antithesis: Richard Stallman launched the GNU Project:

- Concern that users could no longer study and modify source code
- Started campaigning for users' freedom
- Main idea: establish an ecosystem of software that doesn't put restrictions on users

1984: Launched the development of the free operating system GNU:

- GNU Compiler Collection (gcc, g++), GNU Emacs, ls, grep, etc. still widely used

1985: FSF (non-profit organization) founded by Stallman

- "Copyleft" was invented and implemented within the GNU General Public License (GPL) in 1989
- Main idea: ensure user's freedom to obtain, use, study, modify source code and require that derivatives remain under same/similar license

[1] https://archive.org/details/ataribooks-the-atari-basic-source-book

[2] GNU mascot, by Aurelio A. Heckert, CC BY-SA 2.0

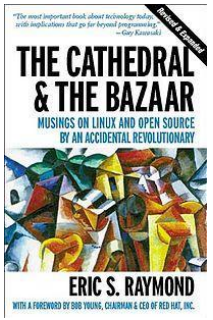
History of FLOSS

1990s: Dot-com boom promoted free software:

- Examples: Use of LAMP (Linux, Apache HTTP Server, MySQL, PHP) as technology stack

1991/1992: Torvalds released Linux kernel (eventually under GNU GPL):

- Establishes GNU/Linux “package” as the first free operating system (“distro”)



1997: Raymond publishes “The Cathedral and the Bazaar”:

- Snapshot/analysis of hacker culture and free software principles
- Netscape releases Communicator suite as free software in 1998

1998: OSI founded as a branch of free-software movement:

- Christine Peterson coins the term “open source”
- People start emphasizing business potential of open source software (rebranding of “free software”)
- Netscape releases Navigator under the Netscape Public License (not GPL compatible!)
- In 2003, Mozilla Foundation was created → parts of Firefox are based on Navigator’s code

[1] Raymond, “The Cathedral & the Bazaar”, O'Reily Media, 2008

[2] Netscape Navigator browser icon used from 1994 to 2007, public domain

History of FLOSS

2000s: Widespread acceptance and establishment:

- Examples: StarOffice (later OpenOffice and LibreOffice)
- Major players in industry consider FLOSS as a business threat:
  - Examples: “Halloween documents” [1], released by Raymond from 1998 to 2004
  - A series of leaks and responses to Microsoft seeing FLOSS and Linux in particular as a threat to its dominance (indirectly acknowledges competitiveness of FLOSS)
  - Note: Microsoft has changed to being a huge contributor to FLOSS!

2000s: Series of lawsuits and court decisions related to FLOSS:

- 2001: United States vs. Microsoft:
  - MS was abusing monopoly power to force pre-installing of Internet Explorer
- 2004-2007: Microsoft vs. European Commission:
  - Anti-competitive practices, integration of Windows Media Player in Windows NT

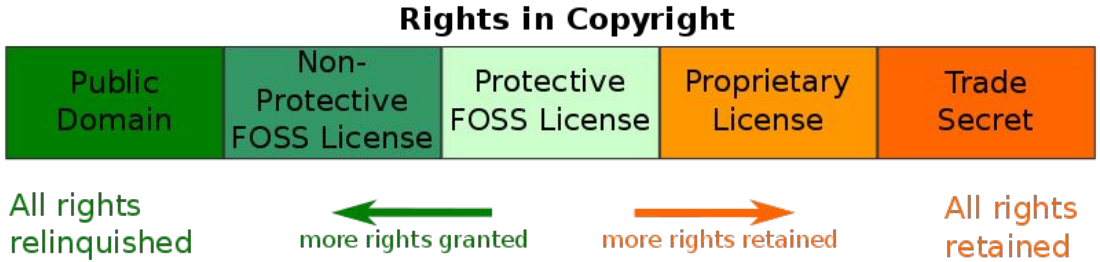
[1] Raymond, “The Halloween Documents”, accessed 2019/10/07

Copyright

- Exclusive but time-limited right of creator to benefit from original work
- Subject to each country's legislation/jurisdiction
- Limitations, such as fair use (US)/fair dealing (UK) provisions :
  - Educational purposes, use of APIs (e.g. Oracle vs. Google), music samples, parody, etc.
- Expiration (typically 50–100 years after creator's death, depending on jurisdiction):
  - Works become public domain after expiration
  - Mickey Mouse copyright extensions (currently until 2023)
- Applies automatically according to "Berne Convention"
- Main driver for developing licenses for open source software

Copyleft vs. copyright

- Copyleft: Goal is to ensure protection of users' freedoms (read, write, modify, distribute)
- Protective (as compared to permissive):
  - Ensures that derivatives (downstream) remain under same terms



- Not only applicable to software (e.g. Creative Commons licenses for content)

[1] Classification of (software) licenses in context of copyright according to Mark Webbink, 2005, public domain
[2] Free Software Foundation, "What is copyleft?", accessed 2019/10/07

Licenses: important candidates

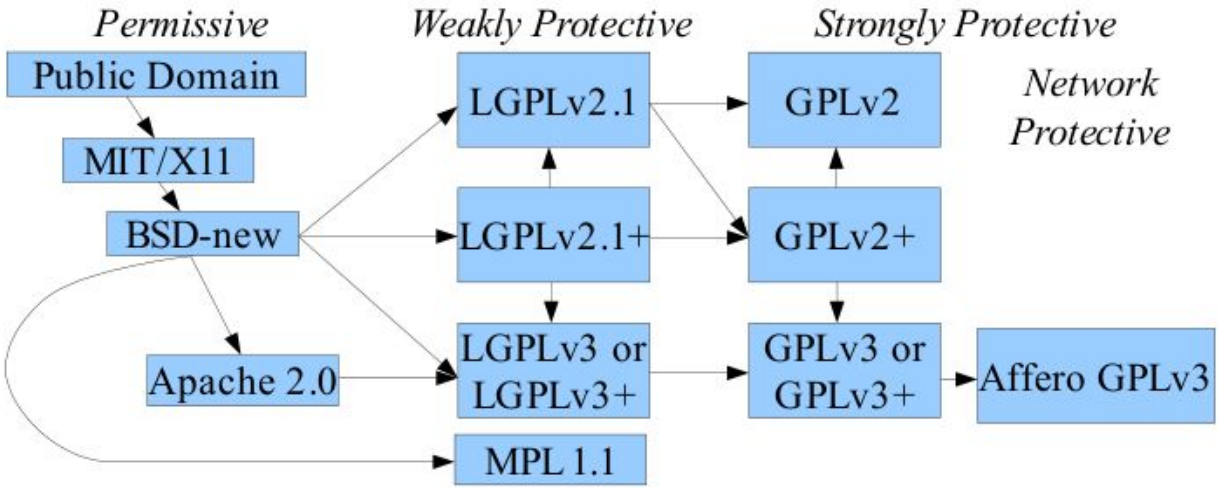
Comparison (see e.g. [2] for more details):

	% of projects [1]	FSF approved	OSI approved	Type
MIT	45%	Yes	Yes	Permissive
GPL v2	13%	Yes	Yes	Copyleft
Apache	11%	Yes	Yes	Permissive
GPL v3	9%	Yes	Yes	Copyleft
BSD	5%	Yes	No (Original)/ Yes (Modified)	Permissive

[1] GitHub Blog, "Open source license usage on github.com", 2015, accessed 2019/10/07
[2] Wikipedia, "Comparison of free and open-source software licenses", accessed 2019/10/03

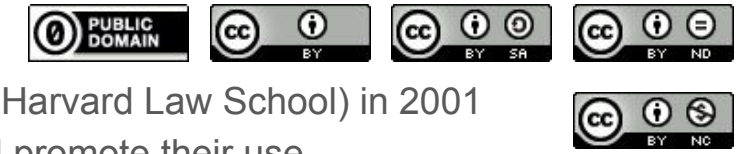
Licenses: compatibilities

- One-way compatibility from permissive to copyleft licenses (see diagram)
- Problematic cases are e.g. combining code, linking libraries, etc.



[1] Compatibility of common licenses, 2007, by David A. Wheeler, CC BY-SA 3.0

What about creative commons?



Creative Commons

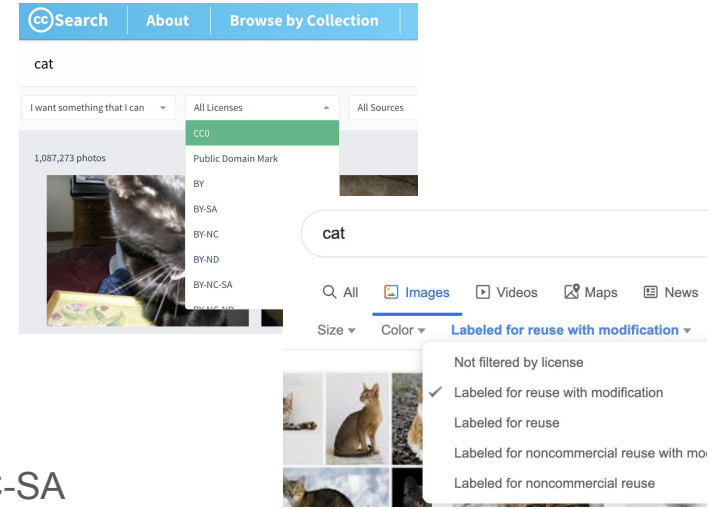
- Co-founded by Lawrence Lessig (professor of Law at Harvard Law School) in 2001
- Goal: create works in public domain to build upon and promote their use
- Introduced a set of licenses
  - Gives creators a simple, standardized way to grant copyright permissions to their creative work
  - Aimed to be internationally valid
- Basic types (+ combinations) [1]
  - CC0: “No Rights Reserved”
  - CC BY: attribution (distribute, remix, tweak, build upon, even commercially)
  - CC BY-SA: share-a-like (new creations under identical terms) → “copyleft”
  - CC BY-ND: no derivatives
  - CC BY-NC: non-commercial use only
- Three-layer design: legal, human readable, machine readable

[1] Creative Commons licenses, “About The Licenses”, accessed 2019/10/08
[2] Wikipedia, “Comparison of free and open-source software licenses”, accessed 2019/10/03

What about creative commons?

Examples of usages (see [1] for a list):

- 2018: 1.4 billion works under CC licenses:
  - 415 million pictures on Flickr under CC license
- PLOS (Public Library of Science):
  - Publishes ~50.000 articles/year under CC BY
- Wikipedia and Wikimedia Commons widely uses CC BY-SA
- Arduino hard/software under CC BY-SA
- Wired.com photography releases photos under CC BY-NC
- OmegaTau podcast publishes episodes under CC BY-NC
- MIT Open CourseWare materials released under CC BY-NC-SA
- TED Talks videos licensed under CC BY-NC-ND



<u>BUT: Creative Commons is not recommended for software!</u>

- Source code distribution, patents, one-way compatibility BY-SA → GPLv3

[1] Wikipedia, "List of major Creative Commons licensed works", accessed 2019/10/07

Issues with open sourced works

Missing access/transparency to know where/how open sourced works are used

Lack of knowledge (software projects, developers, legal issues)

Difficulty of enforcement:

- 2008 FSF vs. Cisco Systems: lawsuit for violating GNU GPL/LGPL:
  - Cisco bought Linksys (vendor of popular WIFI routers)
  - Proprietary firmware contained GPL/LGPL-licensed software, such as GCC
  - Cisco eventually settled, released code as open source, made a financial contribution to FSF

Difficulty of restricting use cases:

- By definition, free software does not allow to exclude use cases: military use, (nuclear) weapons, etc.

[1] Wikipedia, "FSF vs. Cisco Systems", accessed 2019/10/07
[2] Meeker, H. J., "Open Source and The Legend of Linksys", LinusInsider, 2005/06/28
[3] A Linksys wireless-G router, model number WRT54GS by Evan Amos, public domain

Contributors

2017 analysis of GitHub shows that the following companies had the largest number of contributors to open source projects [1]:

1. Microsoft
2. Google
3. Red Hat
4. IBM
5. Intel

Other contributors are volunteers, contractors, students, researchers, etc.

2017 Linux Kernel Development Report:

> "85 percent of all kernel development is demonstrably done by developers who are being paid for their work" [2]

Top companies contributing to the Linux kernel, 4.8–4.13

Company	Changes	%
Intel	10,833	13.1%
none	6,819	8.2%
Red Hat	5,965	7.2%
Linaro	4,636	5.6%
unknown	3,408	4.1%
IBM	3,359	4.1%
consultants	2,743	3.3%
Samsung	2,633	3.2%
SUSE	2,481	3.0%
Google	2,477	3.0%
AMD	2,215	2.7%

[1] Analysis of of public profiles via GitHub REST API, by Fil Maj, https://twitter.com/mjasay/status/960563592683667456
[2] Linux Foundation, "2017 Linux Kernel Development Report", accessed 2019/10/08

Incentives

For companies:

- Monetary:
  - Free to obtain, commercialization
- No need to reinvent the wheel:
  - Allows to modify/adapt
  - Focus on differentiation
- FLOSS software as industry standards:
  - LAMP stack
  - Python
- Community:
  - Patches, bug fixes, issue reports

For individuals:

- IT security (real vs. perceived)
- Altruistic
- Gain experience and knowledge
- Build portfolio
- Community experience

For academics:

- Allows others to use research
- Can be built upon

Open source as business model

- Dual-licensing or Open Core:
  - Develop and offer under multiple licenses: open source + proprietary “enterprise”
  - Offer feature-limited version as FLOSS
  - Examples: Oracle’s MySQL (proprietary and GPLv2)
- Professional services:
  - Training, technical support, consulting, selling executables (compiling and packaging), and implementing features
  - Examples: RedHat, IBM, SUSE
- Advertising-supported software:
  - Examples: Google, Mozilla, Canonical
  - AdBlock Plus whitelisting
  - SourceForge ad banners ($23 million in 2009 [1])

[1] SourceForge Reports Second Quarter Fiscal 2009 Financial Results

Open source as business model

- Software as a service:
  - Offer subscriptions or access to software running on company's infrastructure
- Partnership with funding organizations:
  - Funding provided by governments, universities, companies, foundations.
  - Custom in-house modifications
  - (Research) grants and fellowships:
    - E.g. Google Summer of Code: 2019 approx. 1276 students funded worldwide [1]
    - In Austria: ISPA's netidee.at supports open projects with up to €50,000 (yearly call)
- Donations and branded merchandise:
  - Examples: Mozilla Foundation sells t-shirts, mugs, etc.
- Pre-order/crowdfunding:
  - Examples: Kickstarter (400 projects with tag "open source" [2]), Indiegogo, etc.

[1] Google Summer of Code 2019 (Statistics Part 1)

[2] https://www.kickstarter.com/discover/tags/open-source

Popular tools

- SCM:
  - Git, Mercurial, SVN
- Platforms:
  - CI (build, test coverage, deploy): coverall, travis-ci
  - GitHub, Bitbucket, GitLab
- Documentation:
  - Web pages
  - Wikis
- Issue Tracking:
  - JIRA
- Communication:
  - IRC/Slack channels
  - User/Dev mailing lists

How to contribute

- Find project, check out/fork source code, and familiarize yourself
- Subscribe user/dev mailing lists, monitor community platforms (stackoverflow)
  - Ask questions (IRC), show interest, help other users
- Try to understand the project and its community
  - E.g. release cycles, patches, voting procedures
  - Get to know key figures, e.g. users, contributors, maintainers, mentors, and committers
- Improve/extend/fix features (e.g. by browsing bug issues)
  - Send patches or pull requests
  - Add and improve documentation
- Campaign for the project (e.g. outreach)
- Eventually become active committer

→ Check out open source programs [1], e.g. Google Summer of Code [2]

[1] https://github.com/tapaswenipathak/Open-Source-Programs

[2] https://summerofcode.withgoogle.com/

Open source project metrics

- Number of different committers
- Regularity and size of contributions by committers
- Duration until contributions/pull requests are merged
- Response time to issues
- Number of open issues
- Activity on mailing lists and other communication channels
- Release cycle
- Availability of roadmap
- Availability of documentation

[1] Open-Source-Metriken
[2] Measuring Your Open Source Program's Success

Criticism of open source projects and communities

Accessibility:

> “Despite the rhetoric surrounding Open Source, which basically argues that ‘anybody can contribute,’ it seems instead that only those few participants who have managed to define and present themselves as ‘‘software craftsmen’’ eventually reach the status of developer in a project.” [1]

Inclusion:

> “The open source community needs to show less discrimination and more inclusion to tone down the male-dominated atmosphere into something that promotes participation and not strictly individual work.” [2]

[1] DUCHENEAUT, N. Comput Supported Coop Work (2005) 14: 323. https://doi.org/10.1007/s10606-005-9000-1
[2] Powell, W.E., Hunsinger, D.S., and Medlin, B.D. Gender Differences within the Open Source Community: An Exploratory Study. Journal of Information Technology 21, 4 (2010), 29-37.

Extensions and other uses

- Originally, the term “open source” was applied to software only
- Broader principles and practices around open source software extended to other areas:
  - Open Hardware
  - Open Data
  - Open Content/Open Educational Resources
  - Open Science/Research
  - Open Access
  - Open Spaces/Open Practices

Upcoming tasks

- Next lecture: Open Hardware:
  - **Tuesday, October 15**: 17:00–19:00, Argentinierstraße 8, Seminarraum/Bibliothek 194-05
- Project group forming and topic selection:
  - **Friday, October 25**, via email to **both** lecturers
  - See project ideas at https://free-and-open-technologies.github.io
- First project meeting: week 44/2019: [DAY], [X–Y], seminar room [Z]
- Paper group forming and topic selection:
  - **Friday, November 29**, via email to **both** lecturers

Literature

Bretthauer, D., ”Open Source Software: A History”, 2001

Raymond, E. S., “The Cathedral & the Bazaar”, O'Reily Media, 2008

Von Hippel, E. and von Krogh, G., “Open Source Software and the “Private-Collective” Innovation Model: Issues for Organization Science”, Organization Science (2003) 14 (2)208-223.