

# CS 4530

# Fundamentals of Software Engineering

## Module 16: Open Source Principles

---

Adeel Bhutta, Mitch Wand

Khoury College of Computer Sciences

# Learning Goals

---

By the end of this lesson, you should be able to...

- Understand terminology and explain open source culture and principles
- Opine on philosophical/political debate between open source and proprietary principles
- Reason about tradeoffs of different open source licenses and business model

# Background: laws and open source

---

*Copyright* protects creative, intellectual and artistic works — including software

Alternative: *public domain* (nobody may claim exclusive property rights)

*Trademark* protects the name and logo of a product

OSS is generally copyrighted, with copyright retained by contributors or assigned to entity that maintains it

Copyright holder can grant a *license for use*, placing restrictions on how it can be used (perhaps for a fee)

# Early open source: UNIX to BSD

- Hardware was not yet standardized, computer vendors focused on hardware, building new operating systems for each platform
- Much software development focused in academic labs, and AT&T's Bell Labs
- AT&T is prohibited from entering *new* telecommunications businesses (can't make OS a product)
- Unix created at Bell Labs using the new, portable language "C", licenses initially released with source code
- 1978: UC Berkeley begins distributing their own derived version of Unix (BSD)





# The BSD License is *Permissive*

---

Authors of BSD created a license for the OS that:

1. Required those using it to credit the university (in source, doc, ads)
2. Limited liability for (mis)-use

```
Copyright (c) <year>, <copyright holder> All rights reserved.  
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:  
1.Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.  
2.Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.  
3.All advertising materials mentioning features or use of this software must display the following acknowledgement: This product includes software developed by the <copyright holder>.  
4.Neither the name of the <copyright holder> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.  
THIS SOFTWARE IS PROVIDED BY <COPYRIGHT HOLDER> AS IS AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.... (move waivers of liability)
```

Original BSD license

```
Security policy loaded: Quarantine policy (Quarantine)  
Copyright (c) 1982, 1986, 1989, 1991, 1993  
The Regents of the University of California. All rights reserved.  
MRC Framework successfully initialized  
using 16384 buffer headers and 10240 cluster IO buffer headers  
AppleKeyStore starting (BUILT: Sep 19 2014 00:11:30)
```

BSD Copyright in OS X boot sequence

# UNIX to GNU's Not Unix

---

## Timeline

- 1978: UC Berkeley begins distributing their own derived version of Unix (BSD)
- 1983: AT&T broken up by DOJ, UNIX licensing changed: no more source releases
- Competing commercial vendors all package and sell their derivations of UNIX (AT&T, HP, Sun, IBM, SGI)
- Also 1983: Richard Stallman announces “Starting this Thanksgiving I am going to write a complete Unix-compatible software system called GNU (Gnu’s Not Unix), and give it away free to everyone who can use it”



GNU logo (a gnu wildebeest)



# Free software as a Philosophy

---

“Free as in Speech, not as in beer”

Richard Stallman’s Free Software Foundation — free as in liberties (**GPL**)

- Freedom 0: *run code as you wish, for any purpose*
- Freedom 1: *study how code works, and change it as you wish*
- Freedom 2: *redistributed copies (of original) so you can help others*
- Freedom 3: *distribute copies of your modified version to others*



Richard M Stallman (Licensed under GFDL) 7

# Copyleft v. permissive

---

*Freeware Licenses* “Do whatever you want with this software, but don’t blame me if it doesn’t work” usually fall into two categories:

- *Permissive licenses* (BSD, MIT, Apache) encourage adoption by permitting *combining* OSS with my product, releasing my product under a different license (perhaps not even OS)
- *Copy Left Licenses* (GPL, MPL) “protects the commons” by having all linked code under same license, *transitively requiring more sharing*

Philosophy: *do we force participation, or try to grow/incentivize it in other ways?*



# GNU/Linux (1991-Today)

---

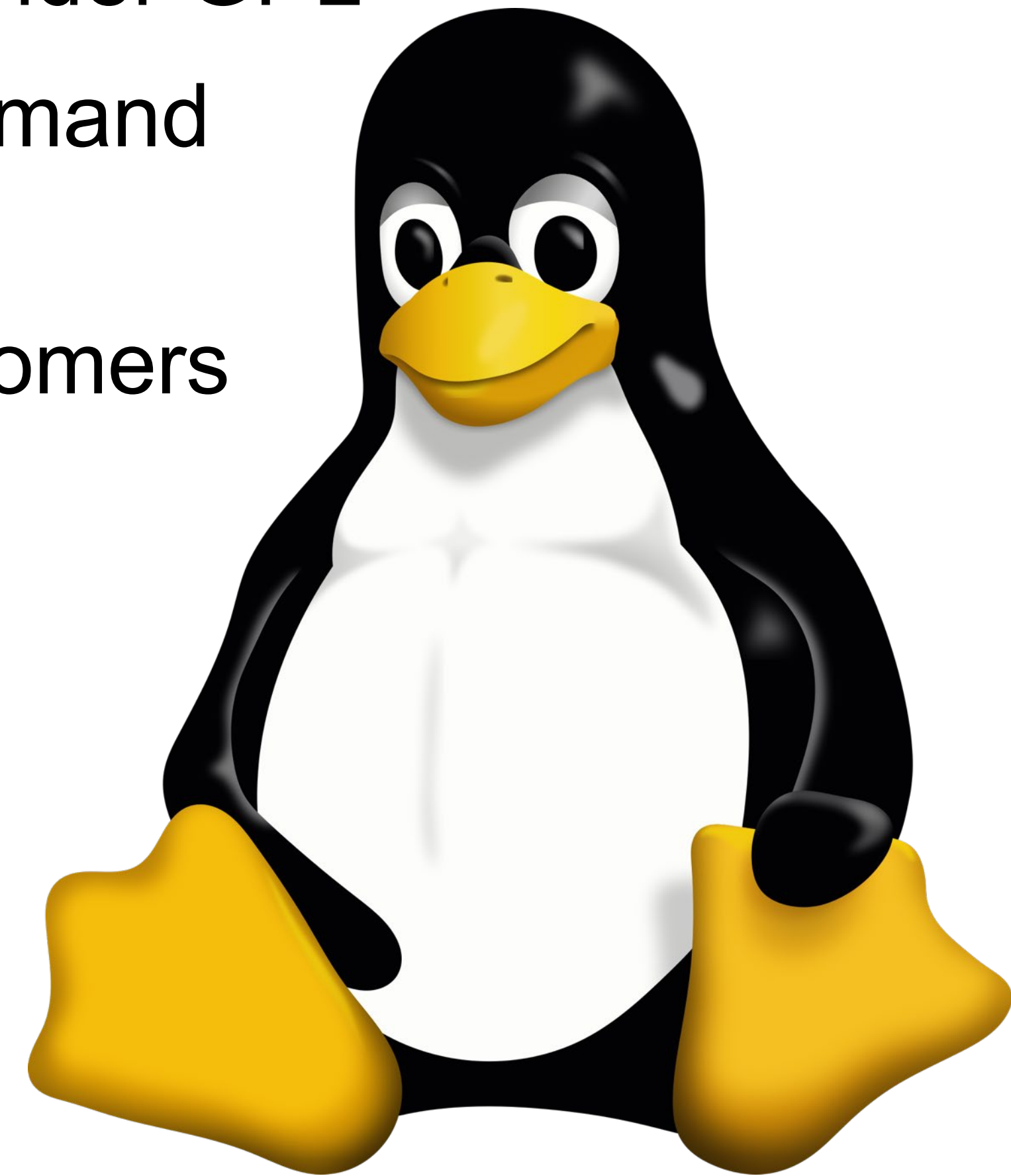
Stallman set out to build an operating system in 1983, ended up building utilities needed by an operating system (compiler, etc)

Linux is built around and with the GNU utilities, licensed under GPL

Rise of the internet, demand for internet servers drives demand for cheap/free OS

Companies adopted and support Linux for enterprise customers

IBM committed over \$1B; Red Hat and others



# Netscape's open-source gambit

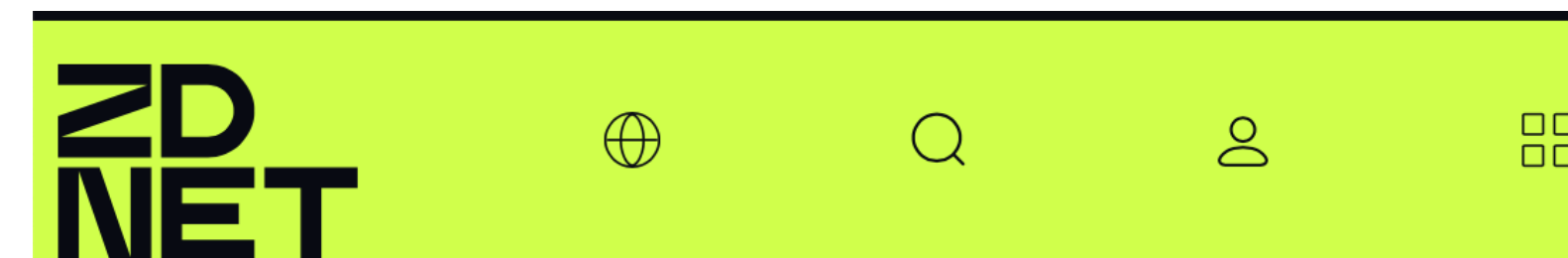
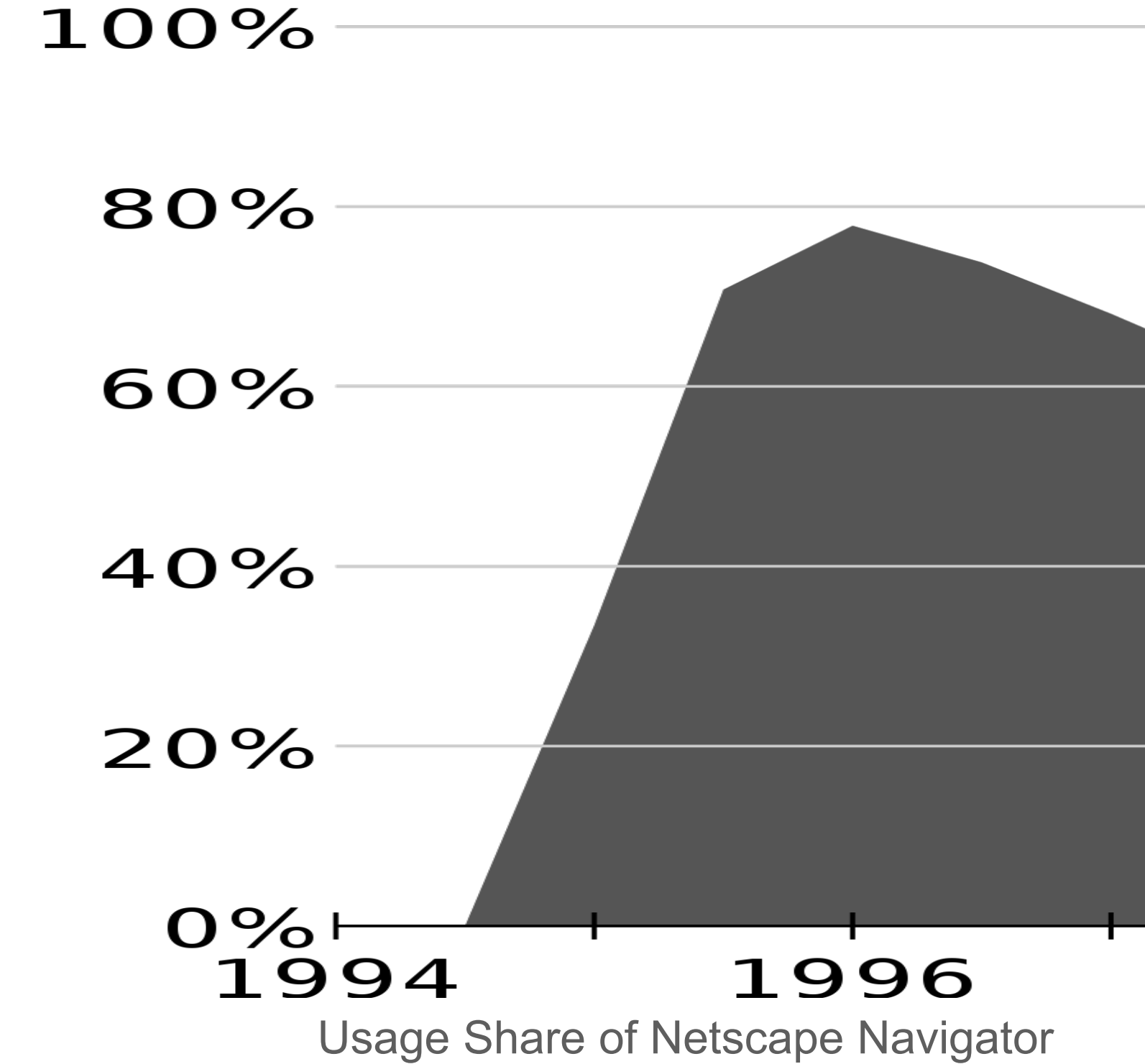
Netscape was dominant web browser early 90's

Business model: free for home and education use, companies pay

Microsoft entered browser market with Internet Explorer, bundled with Windows95, soon overtakes Netscape in usage (free with Windows)

January 1998: Netscape first company to open source code for proprietary product (Mozilla)

- Netscape Public License



Home / Business / Enterprise Software

## Netscape unveils its Navigator source code site

Netscape Communications Corp. is rallying its troops for next month's release of the source code for the company's Navigator Web browser.



# Netscape creates a new license and model

Until Netscape, much of OSS was the FSF and its GPL

**Open Source** coined in 1998 by the Open Source Initiative to capture Netscape's aim for an open development process

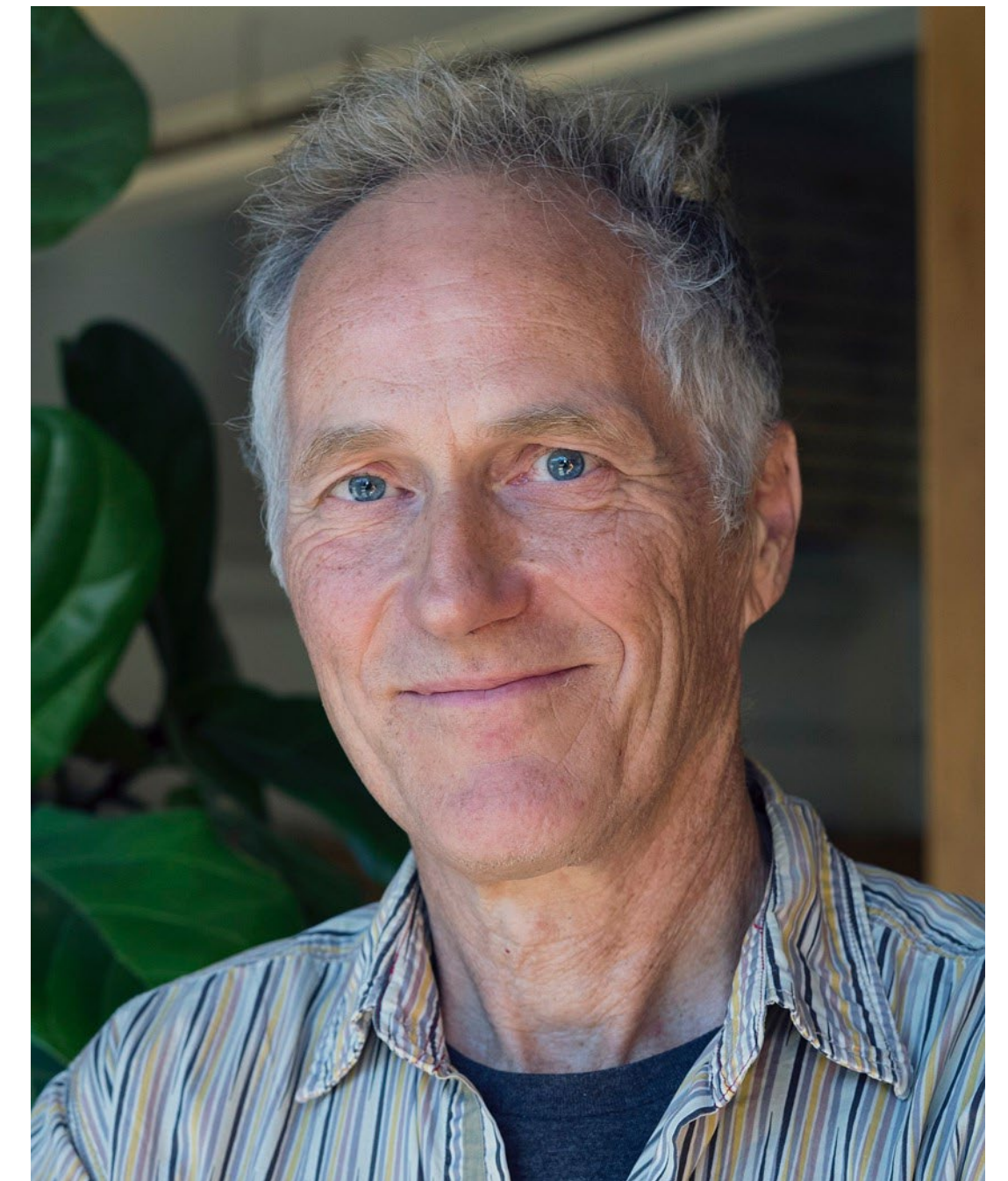
**New licenses** follow, e.g. MIT, Apache, etc. *just like BSD, but without the advertising part*

Publisher Tim O'Reilly organizes a **Freeware Summit** later in 1998, soon rebranded as **Open Source Summit**

*Open Source is a development methodology; free software is a social movement*  
— Richard Stallman



Open source initiative logo



Tim O'Reilly  
Photo via Christopher  
Michel/Flickr, CC BY 2.0



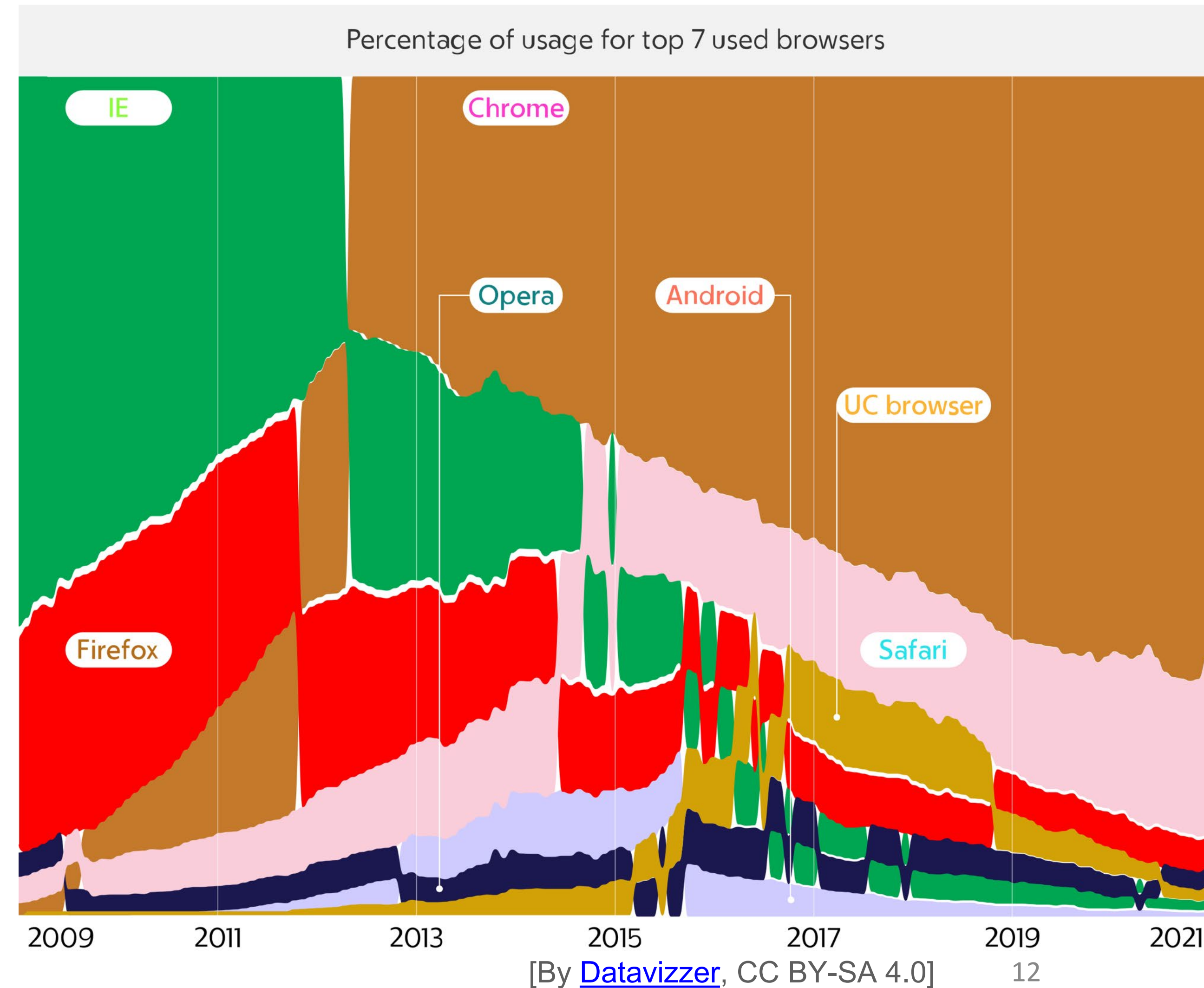
# Firefox lost battle, Open Source wins war

Firefox lost to Chrome and Safari, but OSS won

- Chrome's core = Chromium OSS
  - uses "Blink" rendering engine forked from WebKit
- Safari's core = Webkit OSS
  - forked from KHTML and KJS
- Microsoft's Edge core = Chromium

How do browsers differentiate?

Why is there more than one?





# OSS Provides Community Infrastructure

Operating Systems *are a utility for the common good*: everybody needs them, nobody wants to bear the cost

Eric S Raymond's 1997 essay compares software development methodologies as a "cathedral" or "bazaar"

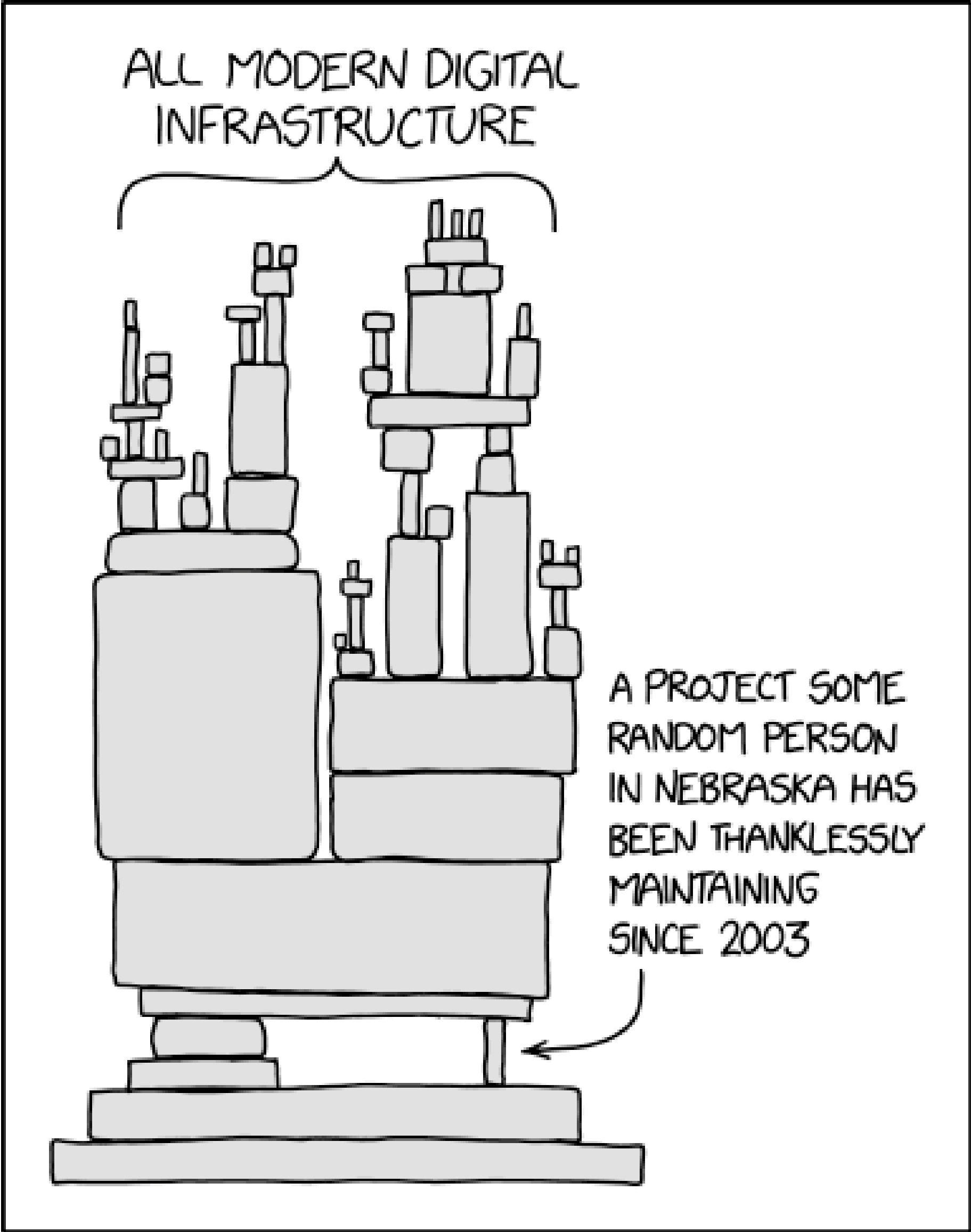
Much OSS today follows "bazaar" model:

- Users treated as co-developers
- Release software early for feedback
- Modularize + reuse components
- Democratic organization





# GitHub is the Modern Bazaar



XKCD "Dependency"



# Adopting OSS has risks

---

Are licenses compatible?

- Including permissive-licensed software in copyleft-licensed software is generally compatible (copyleft takes precedence)

A significant concern for licenses with copyleft: Adopting libraries with copyleft clause generally means what you distribute must also have **same copyleft clause** (and be open source)

- **Are you certain** that the software truly is released under the license that is stated? Did all contributors agree to that license?

# Successful OSS have strong communities

---

OS projects thrive when *community* surrounding them contributes to push the project forward

**Communities** form around collective ownership (even if it's only perceived)

**Contributors** bring more than code: also documentation, support, and outreach

Community/ownership models:

- Corporate owner, community outreach/involvement (MySQL, MongoDB)
- Foundation owner, corporate sponsors (GNU, Linux)

# Open Source Governance

---

Some OSS projects are managed by **for-profit firms**

- **Examples:** Chromium (Google), Moby (Docker), Ubuntu (Canonical), TensorFlow (Google), PyTorch (Meta), Java (Oracle)
- Contributors may be a mix of employees and community volunteers
- Corporations often fund platforms (websites, test servers, deployments, repository hosting, etc.)
- Corporations usually control long-term vision and feature roadmap

Many OSS projects are managed by **non-profit** foundations or ad-hoc communities

- **Examples:** Apache Hadoop/Spark/Hbase/Kafka/Tomcat (ASF), Firefox (Mozilla), Python (PSF), NumPy (community)
- Foundations fund project infrastructure via charitable donations
- Long-term vision often developed via a collaborative process (e.g., Apache) or by benevolent dictators (e.g., Python, Linux)

Business models can support both governance models



# Contributing to open source projects

---

Mature OSS projects often have strict contribution guidelines

- Look for CONTRIBUTING.md or similar

Common requirements:

- Coding style (recall: linters) and passing static checks
- Inclusion of test cases with new code
- Minimum number of code reviews from core devs
- Standards for documentation
- Contributing licensing agreements
  - Without this, you own the copyright and IP for even small bug fixes and that can cause them legal headaches in the future

# When communities move on: Forks

---

The only rights an OSS creator can realistically retain are trademarks on name  
Code will be *forked*

Example:

- 1999: Sun buys StarOffice, GPL'ed as OpenOffice to fight MS Office
- 2010: Oracle buys Sun, fires internal developers, frustrating community
- 2011: Community fork it as LibreOffice, OpenOffice dies (Oracle gifts it to Apache)



# Is Open Source a business model?

February 3, 1976

## An Open Letter to Hobbyists

To me, the most critical thing in the hobby market right now is the lack of good software courses, books and software itself. Without good software and an owner who understands programming, a hobby computer is wasted. Will quality software be written for the hobby market?

Almost a year ago, Paul Allen and myself, expecting the hobby market to expand, hired Monte Davidoff and developed Altair BASIC. Though the initial work took only two months, the three of us have spent most of the last year documenting, improving and adding features to BASIC. Now we have 4K, 8K, EXTENDED, ROM and DISK BASIC. The value of the computer time we have used exceeds \$40,000.

The feedback we have gotten from the hundreds of people who say they are using BASIC has all been positive. Two surprising things are apparent, however. 1) Most of these "users" never bought BASIC (less than 10% of all Altair owners have bought BASIC), and 2) The amount of royalties we have received from sales to hobbyists makes the time spent of Altair BASIC worth less than \$2 an hour.

Why is this? As the majority of hobbyists must be aware, most of you steal your software. Hardware must be paid for, but software is something to share. Who cares if the people who worked on it get paid?

Is this fair? One thing you don't do by stealing software is get back at MITS for some problem you may have had. MITS doesn't make money selling software. The royalty paid to us, the manual, the tape and the overhead make it a break-even operation. One thing you do do is prevent good software from being written. Who can afford to do professional work for nothing? What hobbyist can put 3-man years into programming, finding all bugs, documenting his product and distribute for free? The fact is, no one besides us has invested a lot of money in hobby software. We have written 6800 BASIC, and are writing 8080 APL and 6800 APL, but there is very little incentive to make this software available to hobbyists. Most directly, the thing you do is theft.

What about the guys who re-sell Altair BASIC, aren't they making money on hobby software? Yes, but those who have been reported to us may lose in the end. They are the ones who give hobbyists a bad name, and should be kicked out of any club meeting they show up at.

I would appreciate letters from any one who wants to pay up, or has a suggestion or comment. Just write me at 1180 Alvarado SE, #114, Albuquerque, New Mexico, 87108. Nothing would please me more than being able to hire ten programmers and deluge the hobby market with good software.

*Bill Gates*

Bill Gates  
General Partner, Micro-Soft

The Register

## MS' Ballmer: Linux is communism

After a short silence, Motormouth is back, folks...

[Graham Lea](#)

Mon 31 Jul 2000 // 10:10 UTC

**MS ANALYSTS** Steve Ballmer was the only person to raise the issue of Linux when he wrapped up Microsoft's annual financial analysts meeting in Seattle, although he put Sun and Oracle ahead in terms of being stronger competitors. They of course are 'civilised' competitors - but the Linux crowd, in the world of Prez Steve, are communists.

## Redmond top man Satya Nadella: 'Microsoft LOVES Linux'

Open-source 'love' fairly runneth over at cloud event



The New York Times

## Microsoft Buys GitHub for \$7.5 Billion, Moving to Grow in Coding's New Era

Give this article



A GitHub billboard being installed in San Francisco in 2014. Microsoft said on Monday that it would acquire the company for \$7.5 billion. David Paul Morris/Bloomberg



# “Open Source as a Utility” is a common model

The largest, most successful open source projects implement utility infrastructure:

- Operating systems, web servers, logging libraries, programming languages
- **Business model:** build and sell products and services using those utilities, contribute improvements back to the ecosystem
  - Linux, Kubernetes, React, etc.
- Many companies provide specialized *distributions* of these open source infrastructure and specialized tools to improve them; support upstream project



**Red Hat**



# Monetize OSS with “open core, closed plugins”

Model: **core component** of product is open source; **plugins** for a fee

Example: Apache Kafka, a distributed message broker (glue in event-based system)

- Product is open source, maintained by Apache foundation, supported by a company
- Confluent provides plugins to connect Kafka to different systems out-of-the-box



# Monetize OSS with dual licenses (?)

**Model:** Offer a free copyleft license to encourage adoption, prevent competitors from improving it without sharing improvements.

Offer custom, more permissive licenses to third parties willing to pay for that (needed to bundle MySQL with a proprietary system)

Only possible when there is a single copyright owner, who can unilaterally change license

Risk of losing control of the copyleft portion: nothing to stop the community from forking it

- MariaDB was forked from MySQL





# Monetize OSS by selling it as a SaaS (?)

**Model:** Creators of OSS provide a cloud hosted, “fully managed” installation as a service

**Risk:** What is your competitive advantage over cloud utility providers?

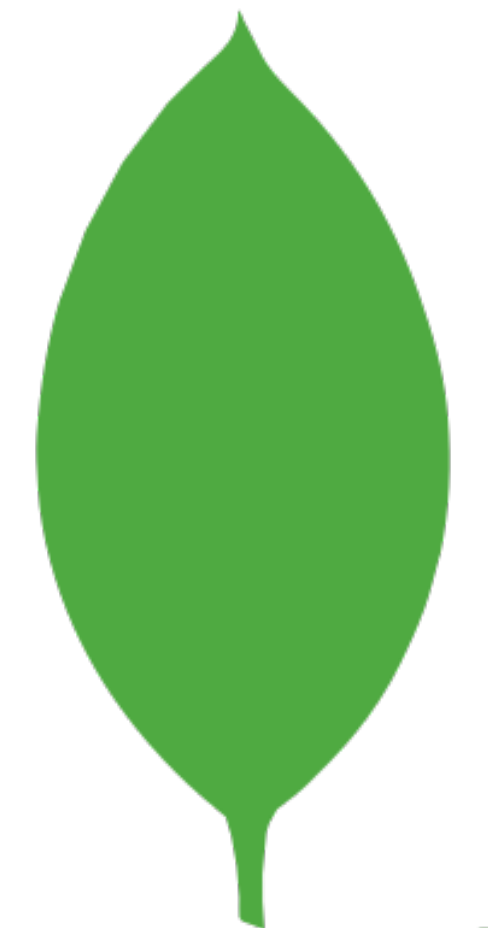
- Amazon improves your GPL code without sharing because it is not distributing it (operates it as a service)

**Example:** MongoDB Atlas (document-oriented database)

- MongoDB created a new copyleft license for providers operating MongoDB as a service
- Amazon forked GPL'ed MongoDB



**Amazon DocumentDB**



**mongoDB**

# Monetize OSS by offering title insurance

---

SQLite is an extremely popular database library, *in the public domain* (copyright is waived)

- License text:

The author disclaims copyright to this source code. In place of a legal notice, here is a blessing:

May you do good and not evil.

May you find forgiveness for yourself and forgive others.

May you share freely, never taking more than you give.

- To have *legal proof* that the code is in the public domain (traceable links from code to waivers of copyright), you pay money



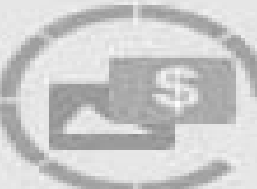

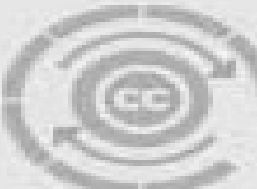

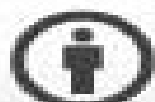












# Other Licenses


<https://creativecommons.org/share-your-work/cclicenses/>

- CC Licenses allow both copy-left and permissive

**CREATIVE COMMONS LICENSES**

		 COPY & PUBLISH	 ATTRIBUTION REQUIRED	 COMMERCIAL USE	 MODIFY & ADAPT	 CHANGE LICENSE
	PUBLIC DOMAIN	✓	✗	✓	✓	✓
	CC BY	✓	✓	✓	✓	✓
	CC BY-SA	✓	✓	✓	✓	✗
	CC BY-ND	✓	✓	✓	✗	✓
	CC BY-NC	✓	✓	✗	✓	✓
	CC BY-NC-SA	✓	✓	✗	✓	✗
	CC BY-NC-ND	✓	✓	✗	✗	✓

 You can redistribute (copy, publish, display, communicate, etc.)
  You have to attribute the original work
  You can use the work commercially
  You can modify and adapt the original work
  You can choose license type for your adaptations of the work.

 Creative Commons: The Ultimate Guide by foter.com is licensed under a Creative Commons Attribution-ShareAlike 3.0 United States License. Based on a work at <http://bit.ly/1eWg7W3>



# Other philosophies are expressed in some licenses



# OSS and JSLint → ESLint

---

- JSLint was written by Douglas Crockford for checking if JavaScript source code complies with coding rules (2002)
  - Original License: MIT with modification
  - Changed to FSF / OSI approved “Unlicense” license (public domain) – 2021
- Anton Kovalyov forked it to create JSHint (2011)
- Nicholas C. Zakas created ESLint (2013), where all rules are configurable, and additional rules can be defined or loaded at run-time (Zakas worked on JSHint project)
- Palantir developed TSLint (2013) which was later deprecated
- Marat Dulin created JSCS (2014). In 2016, the JSCS Team joined the ESLint project and has since discontinued maintenance of the JSCS tool.
- OSS development methodology lives on ...



# Learning Goals

---

You should be able to...

- Understand terminology and explain open source culture and principles
- Opine on philosophical/political debate between open source & proprietary principles
- Reason about tradeoffs of different open source licenses and business model