Open Source

17-313 Spring 2024

Foundations of Software Engineering

https://cmu-313.github.io

Michael Hilton, Austin Henley, and Nadia Nahar

Administrivia

- P4 Clarification
  - Grading will be adjusted to account for technical difficulties.
- Final Exam attendance Mandatory:
  - Monday, Monday, May 5, 2025 01:00pm - 04:00pm
  - Location TBD

Learning Goals

- Distinguish between open-source software, free software, and commercial software.
- Identify the common types of software licenses and their implications.
- Distinguish between copyright and intellectual property.
- Express an educated opinion on the philosophical/political debate between open source and proprietary principles.
- Describe how open-source ecosystems work and evolve, in terms of maintainers, community contribution, and commercial backing
- Identify various concerns of commercial entities in leveraging open-source, as well as strategies to mitigate these.

Early(ish) Course Feedback

Keep Doing

- Strong TA feedback
- In class activities
- Candy
- Group work
- Team evaluation forms
- Stories
- Enthusiasm

Start Doing

- SE Workplace dynamics contents
- Faster feedback
- More technical (coding) instruction
- More Details in the assignments (also TLDR for the assignments)
- More Dog pictures
- More NodeBB intro
- More days of office hours
- Releasing assignments earlier

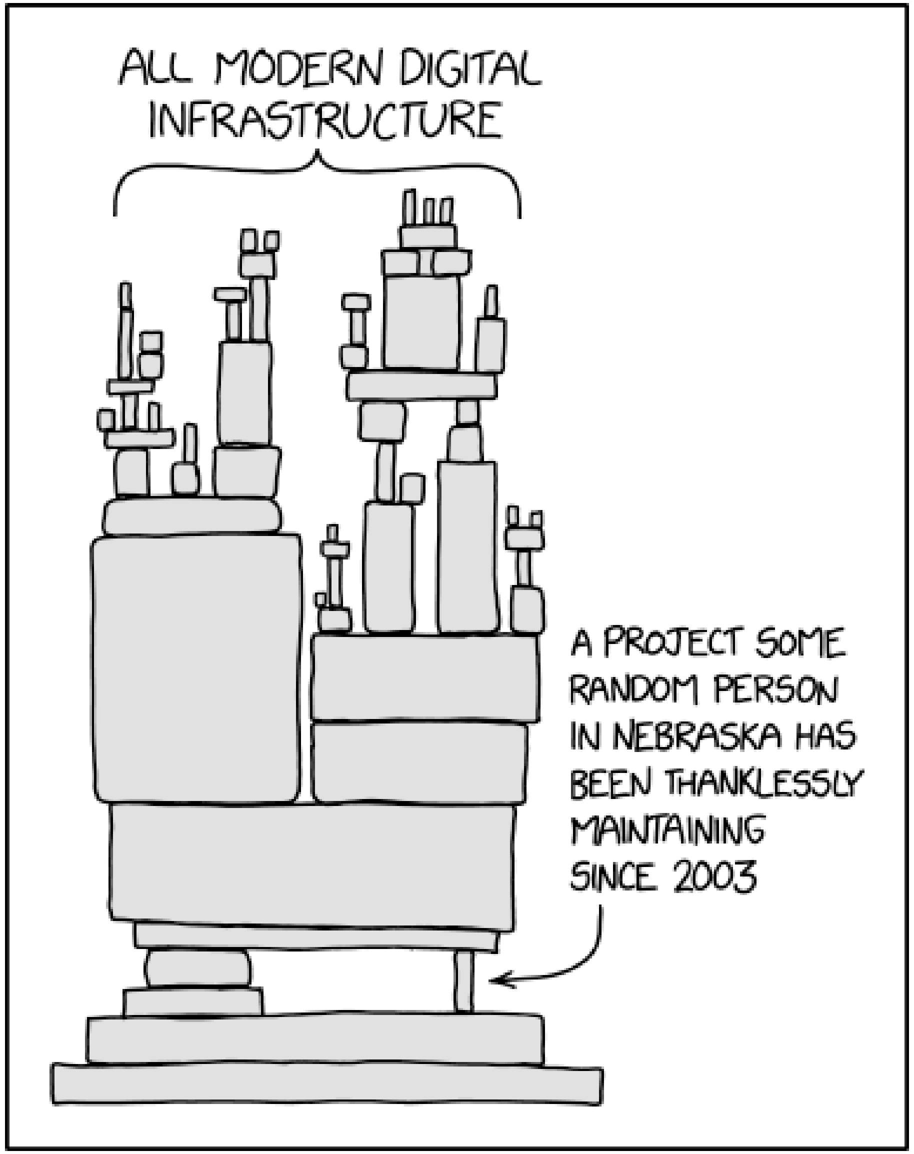
Stop Doing

- So many case studies
- NodeBB (several comments on this)
- Teamwork quizzes on Canvas (they should be on gradescope)
- Exams and attendance

Open Source

Background: laws and open source

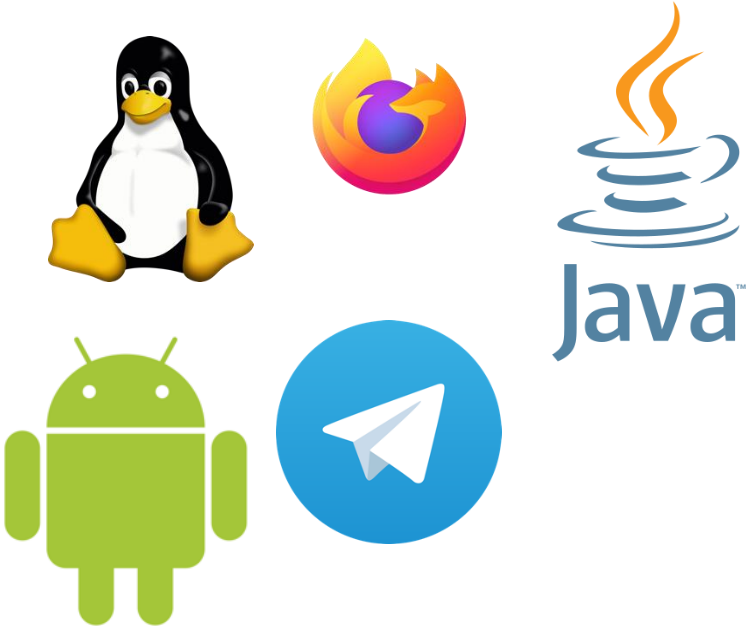
- *Copyright* protects creative, intellectual and artistic works — including software
- Alternative: *public domain* (nobody may claim exclusive property rights)
- *Trademark* protects the name and logo of a product
- OSS is generally copyrighted, with copyright retained by contributors or assigned to entity that maintains it
- Copyright holder can grant a *license for use*, placing restrictions on how it can be used (perhaps for a fee)


ALL MODERN DIGITAL INFRASTRUCTURE
A PROJECT SOME RANDOM PERSON IN NEBRASKA HAS BEEN THANKLESSLY MAINTAINING SINCE 2003

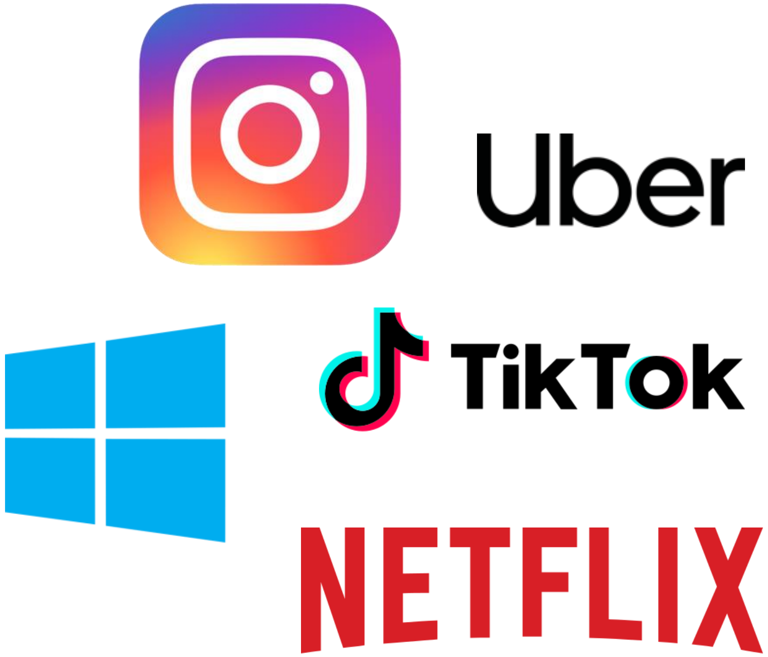
https://xkcd.com/2347/

What is Open-Source Software?

Open-source



Proprietary



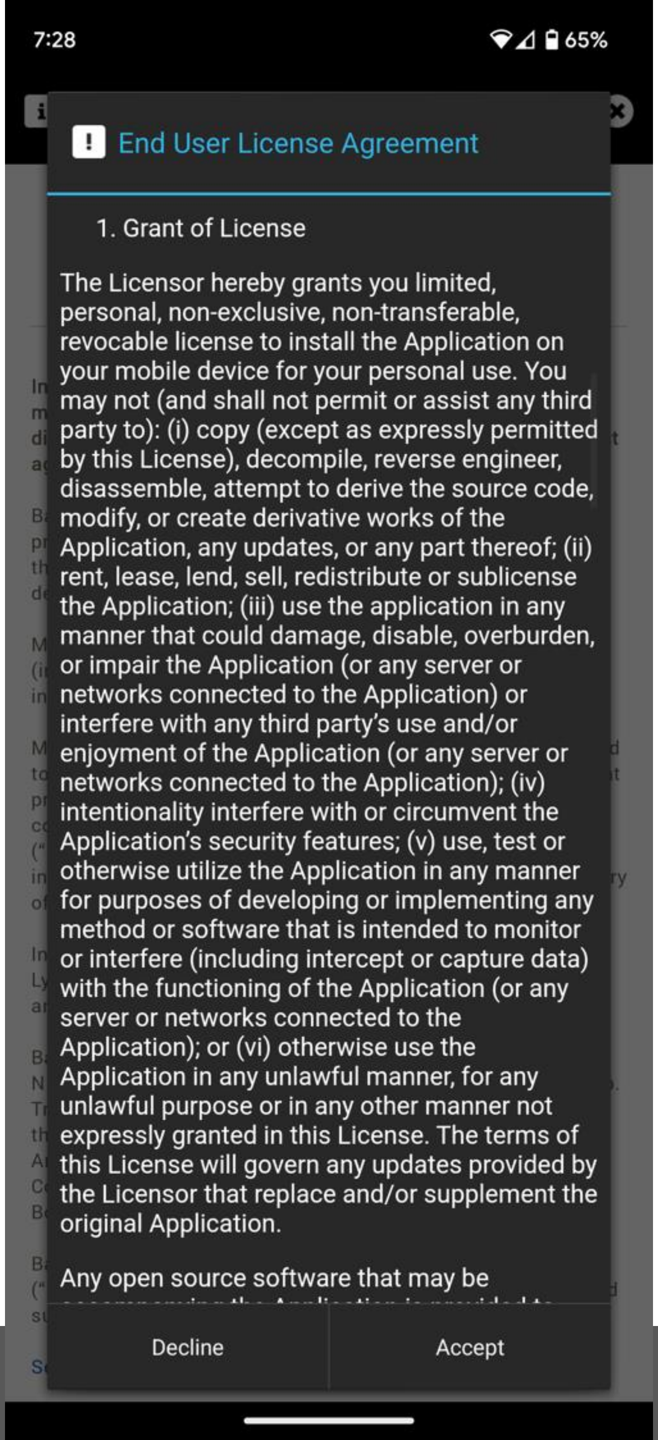
What is Open-Source Software (OSS)?

- Source code availability
- Right to modify and creative derivative works
- (Often) Right to redistribute derivate works

Contrast with proprietary software: a black box

- Intention is to be used, not examined, inspected, or modified.
- No source code – only download a binary (e.g., an app) or use via the internet (e.g., a web service).
- Often contains an End User License Agreement (EULA) governing rights and liabilities.
- EULAs may specifically prohibit attempts to understand application internals.

Example: Bank app on my phone

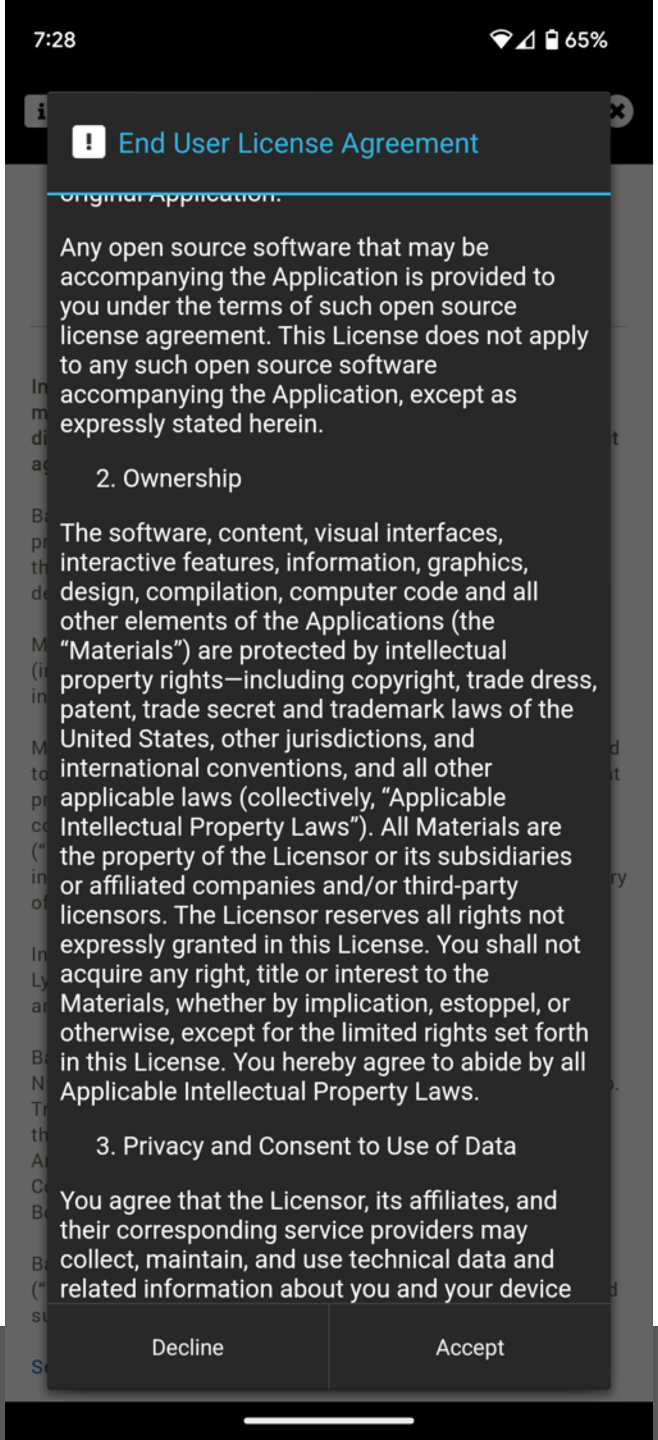


End User License Agreement

1. Grant of License

The Licensor hereby grants you limited, personal, non-exclusive, non-transferable, revocable license to install the Application on your mobile device for your personal use. You may not (and shall not permit or assist any third party to): (i) copy (except as expressly permitted by this License), decompile, reverse engineer, disassemble, attempt to derive the source code, modify, or create derivative works of the Application, any updates, or any part thereof; (ii) rent, lease, lend, sell, redistribute or sublicense the Application; (iii) use the application in any manner that could damage, disable, overburden, or impair the Application (or any server or networks connected to the Application) or interfere with any third party's use and/or enjoyment of the Application (or any server or networks connected to the Application); (iv) intentionality interfere with or circumvent the Application's security features; (v) use, test or otherwise utilize the Application in any manner for purposes of developing or implementing any method or software that is intended to monitor or interfere (including intercept or capture data) with the functioning of the Application (or any server or networks connected to the Application); or (vi) otherwise use the Application in any unlawful manner, for any unlawful purpose or in any other manner not expressly granted in this License. The terms of this License will govern any updates provided by the Licensor that replace and/or supplement the original Application.

Any open source software that may be



End User License Agreement

Any open source software that may be accompanying the Application is provided to you under the terms of such open source license agreement. This License does not apply to any such open source software accompanying the Application, except as expressly stated herein.

2. Ownership

The software, content, visual interfaces, interactive features, information, graphics, design, compilation, computer code and all other elements of the Applications (the "Materials") are protected by intellectual property rights—including copyright, trade dress, patent, trade secret and trademark laws of the United States, other jurisdictions, and international conventions, and all other applicable laws (collectively, "Applicable Intellectual Property Laws"). All Materials are the property of the Licensor or its subsidiaries or affiliated companies and/or third-party licensors. The Licensor reserves all rights not expressly granted in this License. You shall not acquire any right, title or interest to the Materials, whether by implication, estoppel, or otherwise, except for the limited rights set forth in this License. You hereby agree to abide by all Applicable Intellectual Property Laws.

3. Privacy and Consent to Use of Data

You agree that the Licensor, its affiliates, and their corresponding service providers may collect, maintain, and use technical data and related information about you and your device

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
- 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)
- AT&T is prohibited from entering *new* telecommunications businesses (can't make OS a product)

IBM 704 at NASA Langley in 1957 (Public domain)

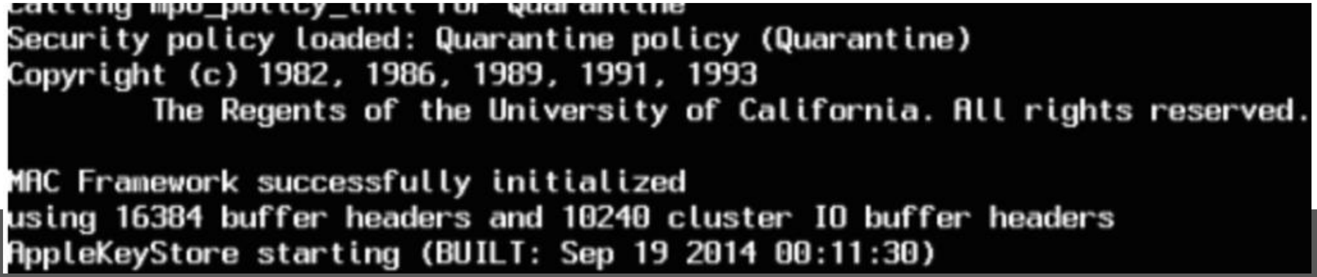
The BSD License is *Permissive*

- Authors of BSD created a license for the OS that:

1. Required those using it to credit the university
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



```
Calling mpo_policy_init for Quarantine
Security policy loaded: Quarantine policy (Quarantine)
Copyright (c) 1982, 1986, 1989, 1991, 1993
        The Regents of the University of California. All rights reserved.

MAC 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: “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

  - 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)

Free software as a Philosophy

- "Free as in Speech, not as in beer"

FSF: software licensed under GNU Public License (GPL), considering questions like:

  - Required to redistribute modifications (under same license)? *Yes, "copyleft"*
  - Can you combine it with more restrictive licenses? *No, not even with BSD!*

Alternative (more like BSD):

"Do whatever you want with this software, but don't blame me if it doesn't work" *freeware*

Copyleft v. permissive

- Can I *combine* OSS with my product, releasing my product under a different license (perhaps not even OS)?
- *Permissive licenses* encourage adoption by permitting this practice
- *Copyleft* “protects the commons” by having all linked code under same license, *transitively requiring more sharing*

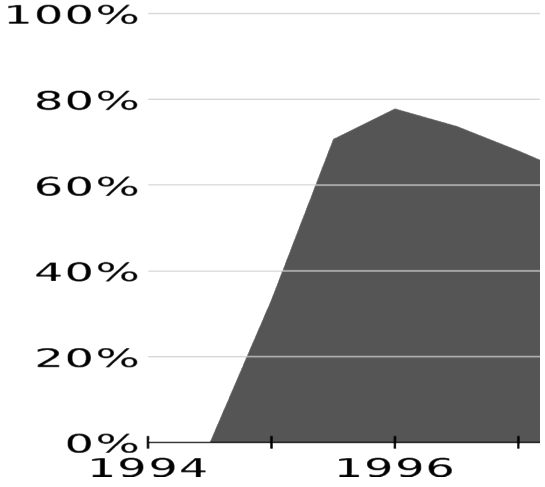
- Philosophy: *do we force participation, or try to grow/incentivize it in other ways?*

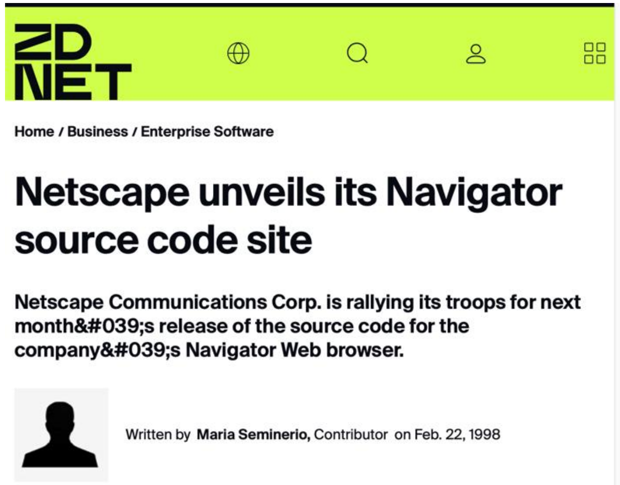
GNU/Linux (1991-Today)

- 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 custome
- 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)





Home / Business / Enterprise Software

Netscape unveils its Navigator source code site

Netscape Communications Corp. is rallying its troops for next month&#039;s release of the source code for the company&#039;s Navigator Web browser.

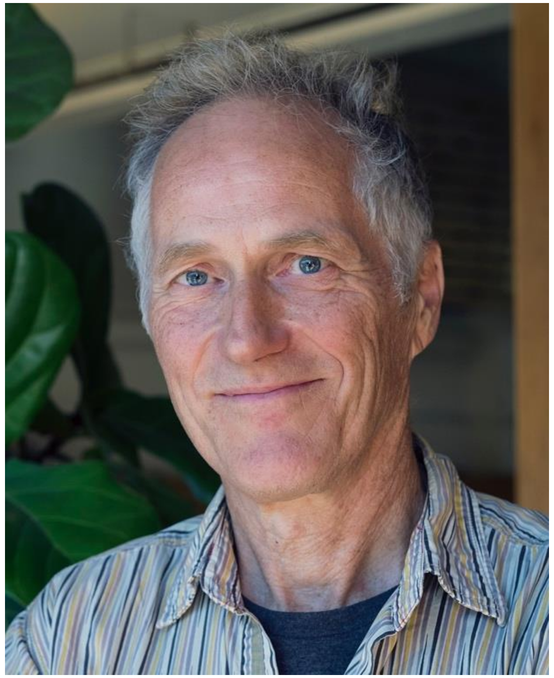
Written by **Maria Seminerio,** Contributor on Feb. 22, 1998

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

WHEN YOU PROGRAM OPEN SOURCE,
YOU'RE PROGRAMMING
COMMUNISM
A REMINDER
from
YOUR FRIENDS AT MICROSOFT

Perception (from some):

- Anarchy
- Demagoguery
- Ideology
- Altruism

Why Go Open Source (vs. Proprietary) ?

Advantages

Disadvantages

Why Go Open Source (vs. Proprietary) ?

Advantages

- Transparency, gain user trust
- Many eyes: crowd-source bug reports and fixes
- Security: more likely for vulnerabilities to be quickly identified
- Community and adoption: get others to contribute features, build stuff around you, or fork your project

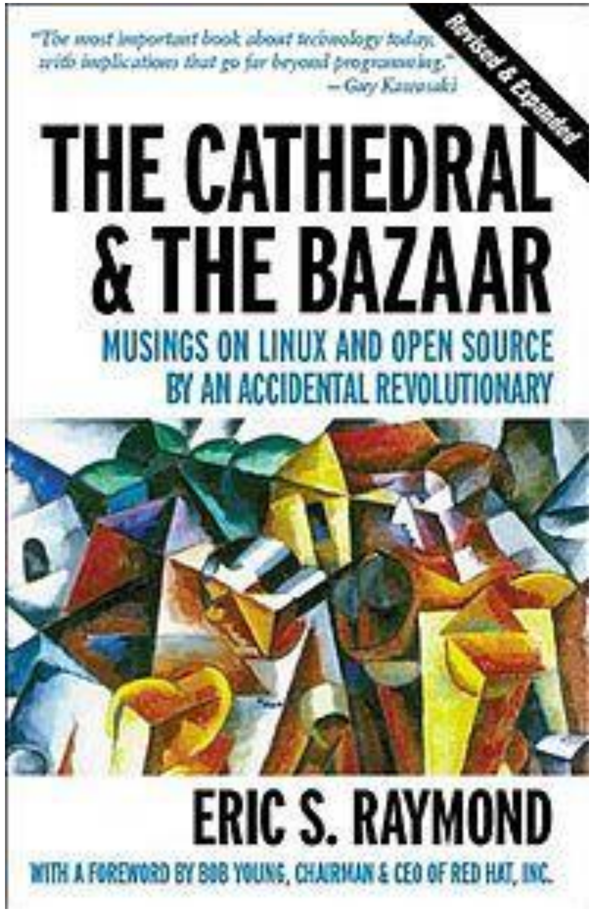
Disadvantages

- Reveal implementation secrets
- Many eyes: users can find faults more easily
- Security: more likely for others to find vulnerabilities first
- Control: You may not be able to influence the long-term direction of your platnform

Open-Source Ecosystems

How OSS is developed

The Cathedral and the Bazaar


"The most important book about technology today, with implications that go far beyond programming."
— Guy Kawasaki
Revised & Expanded
THE CATHEDRAL & THE BAZAAR
MUSINGS ON LINUX AND OPEN SOURCE BY AN ACCIDENTAL REVOLUTIONARY
ERIC S. RAYMOND
WITH A FOREWORD BY BOB YOUNG, CHAIRMAN & CEO OF RED HAT, INC.

The Bazaar won

Cathedral

- Developed centrally by a core group of members
- Available for all once complete (or at releases)
- Examples: GNU Emacs, GCC (back in the 1990s)
- “Sort-of” examples today: Chrome, IntelliJ

Bazaar

- Developed openly and organically
- Wide participation (in theory, anyone can contribute)
- Examples: Linux

OSS has many stakeholders / contributors

- Core members
  - Often (but not always) includes the original creators
  - Direct push access to main repository
  - May be further split into admin roles and developers
- External contributors
  - File bug reports and report other issues
  - Contribute code and documentation via pull requests
- Other supporters
  - Beta testers (users)
  - Sponsors (financial or platform)
  - Steering committees or public commenters (for standards and RFCs)
- Spin-offs
  - Maintainers of forks of the original repository

Contributing processes

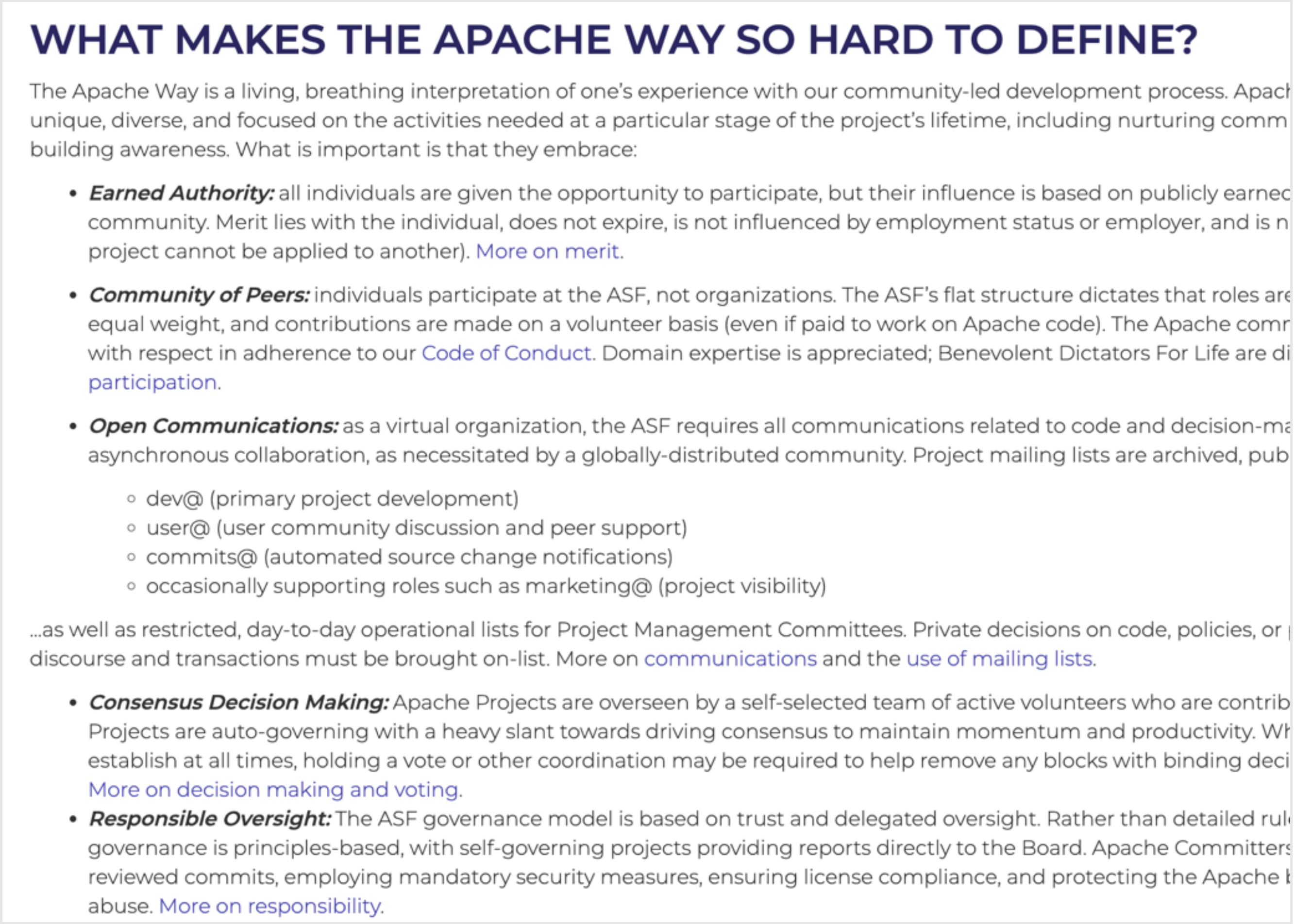
- Mature OSS projects often have strict contribution guidelines
  - Look for CONTRIBUTING.md or similar

- 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 (more on that later)

Governence

- 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)
- Corporations still heavily rely on community-owned OSS projects
  - Many OSS non-profits are funded by Big Tech (e.g., Mozilla by Google)

Example: Apache



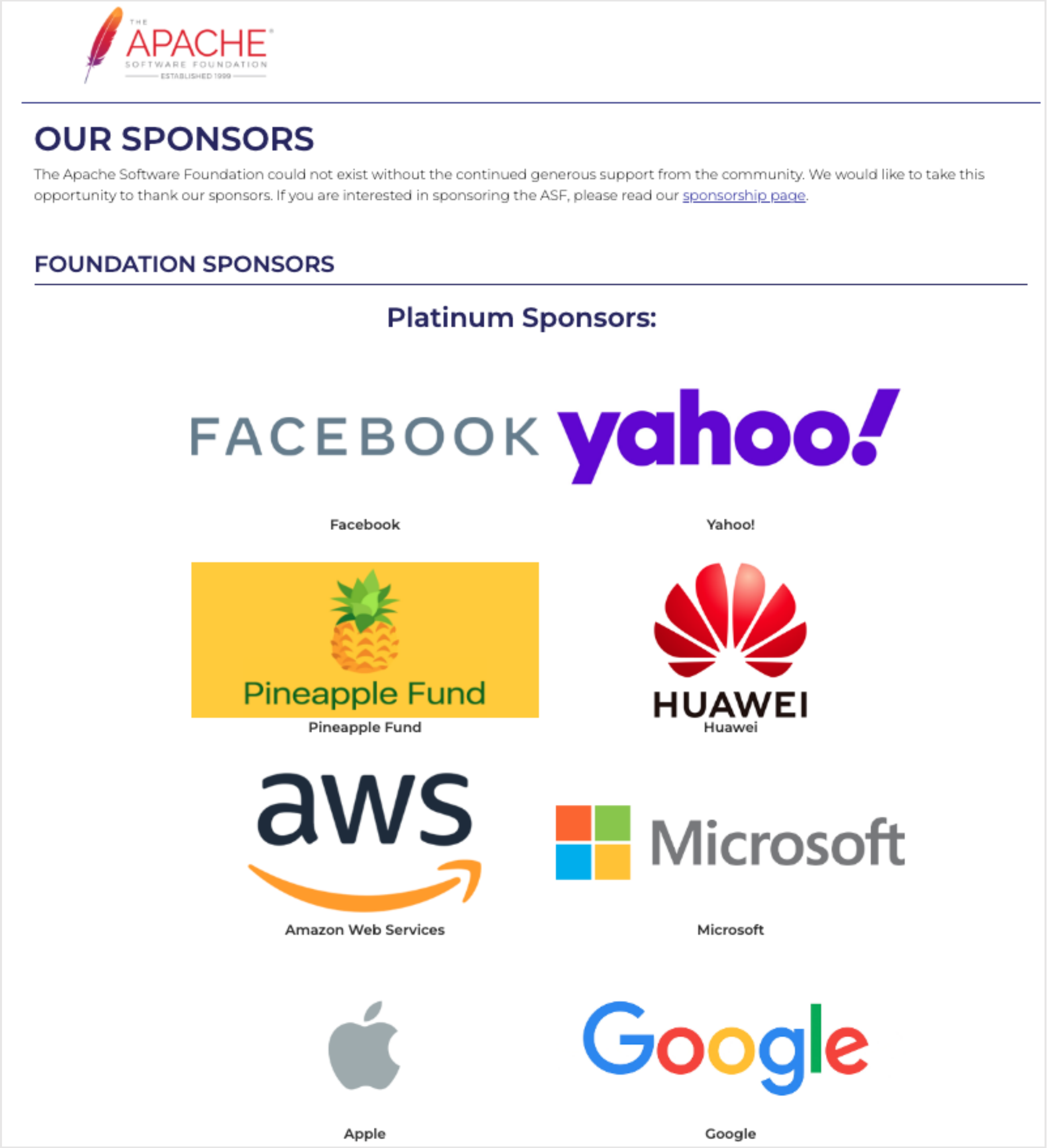
WHAT MAKES THE APACHE WAY SO HARD TO DEFINE?

The Apache Way is a living, breathing interpretation of one's experience with our community-led development process. Apach unique, diverse, and focused on the activities needed at a particular stage of the project's lifetime, including nurturing comm building awareness. What is important is that they embrace:

- ***Earned Authority:*** all individuals are given the opportunity to participate, but their influence is based on publicly earnec community. Merit lies with the individual, does not expire, is not influenced by employment status or employer, and is n project cannot be applied to another). More on merit.
- ***Community of Peers:*** individuals participate at the ASF, not organizations. The ASF's flat structure dictates that roles are equal weight, and contributions are made on a volunteer basis (even if paid to work on Apache code). The Apache comr with respect in adherence to our Code of Conduct. Domain expertise is appreciated; Benevolent Dictators For Life are di participation.
- ***Open Communications:*** as a virtual organization, the ASF requires all communications related to code and decision-ma asynchronous collaboration, as necessitated by a globally-distributed community. Project mailing lists are archived, pub
    - dev@ (primary project development)
    - user@ (user community discussion and peer support)
    - commits@ (automated source change notifications)
    - occasionally supporting roles such as marketing@ (project visibility)

...as well as restricted, day-to-day operational lists for Project Management Committees. Private decisions on code, policies, or discourse and transactions must be brought on-list. More on communications and the use of mailing lists.

- ***Consensus Decision Making:*** Apache Projects are overseen by a self-selected team of active volunteers who are contrib Projects are auto-governing with a heavy slant towards driving consensus to maintain momentum and productivity. Wh establish at all times, holding a vote or other coordination may be required to help remove any blocks with binding deci More on decision making and voting.
- ***Responsible Oversight:*** The ASF governance model is based on trust and delegated oversight. Rather than detailed rul governance is principles-based, with self-governing projects providing reports directly to the Board. Apache Committers reviewed commits, employing mandatory security measures, ensuring license compliance, and protecting the Apache abuse. More on responsibility.



OUR SPONSORS

The Apache Software Foundation could not exist without the continued generous support from the community. We would like to take this opportunity to thank our sponsors. If you are interested in sponsoring the ASF, please read our sponsorship page.

FOUNDATION SPONSORS

Platinum Sponsors:

Facebook

Yahoo!

Pineapple Fund

Huawei

Amazon Web Services

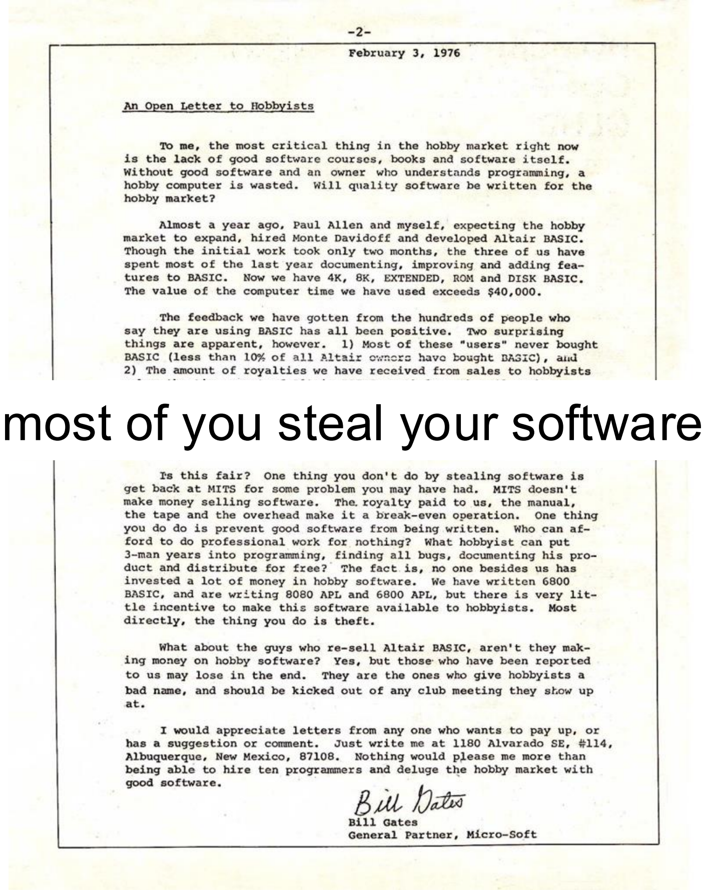
Microsoft

Apple

Google

https://www.apache.org/theapacheway/

Corporate outlook towards open-source has evolved over the years



-2-

February 3, 1976

<u>An Open Letter to Hobbyists</u>

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

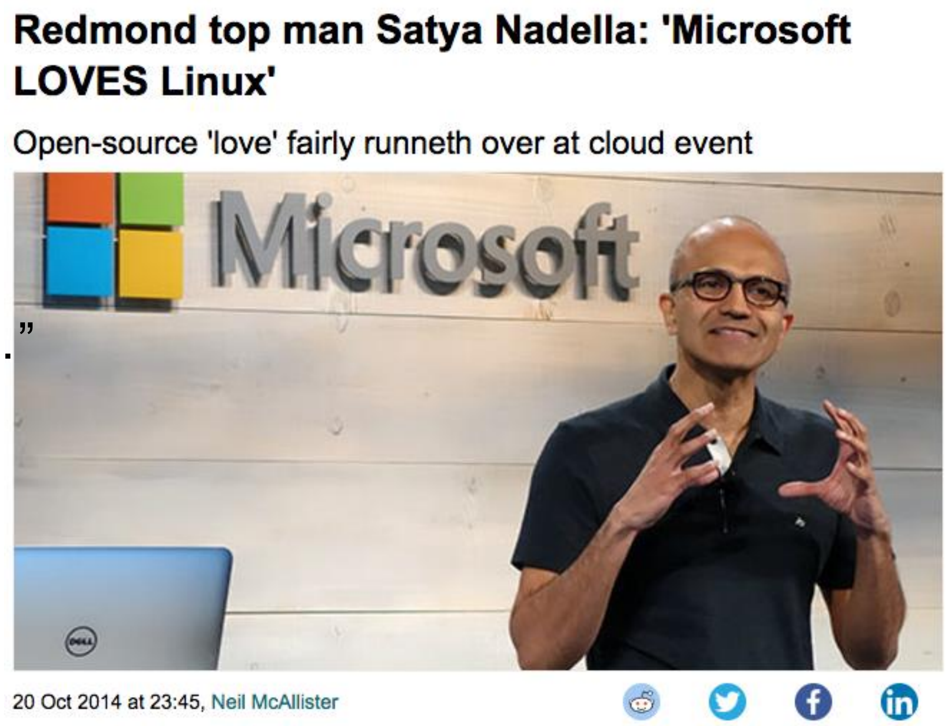
"…most of you steal your software…"

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

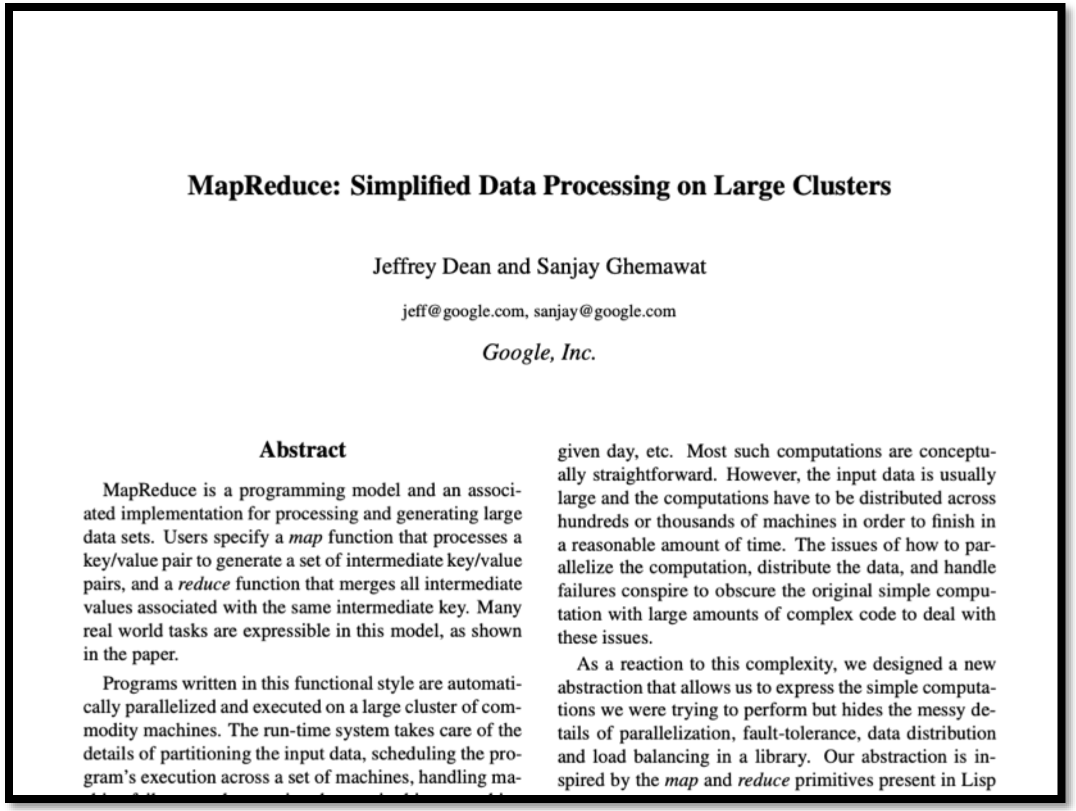


Redmond top man Satya Nadella: 'Microsoft LOVES Linux'

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

20 Oct 2014 at 23:45, Neil McAllister

Risks in *not* open-sourcing?



MapReduce: Simplified Data Processing on Large Clusters

Jeffrey Dean and Sanjay Ghemawat

jeff@google.com, sanjay@google.com

Google, Inc.

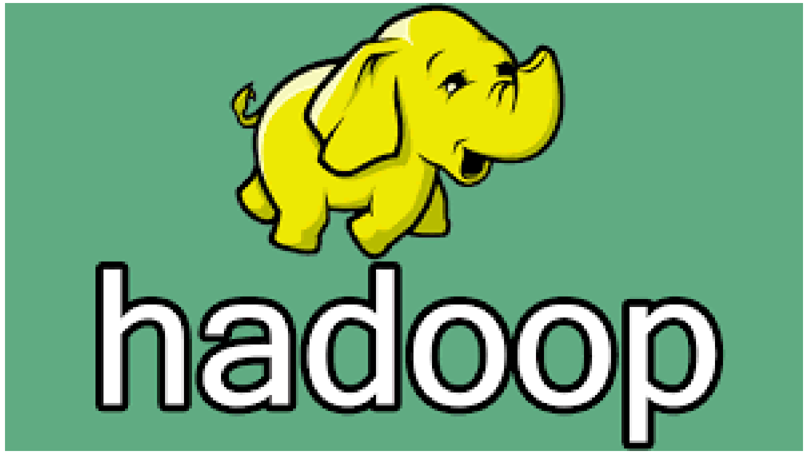
Abstract

MapReduce is a programming model and an associated implementation for processing and generating large data sets. Users specify a *map* function that processes a key/value pair to generate a set of intermediate key/value pairs, and a *reduce* function that merges all intermediate values associated with the same intermediate key. Many real world tasks are expressible in this model, as shown in the paper.

Programs written in this functional style are automatically parallelized and executed on a large cluster of commodity machines. The run-time system takes care of the details of partitioning the input data, scheduling the program's execution across a set of machines, handling ma-

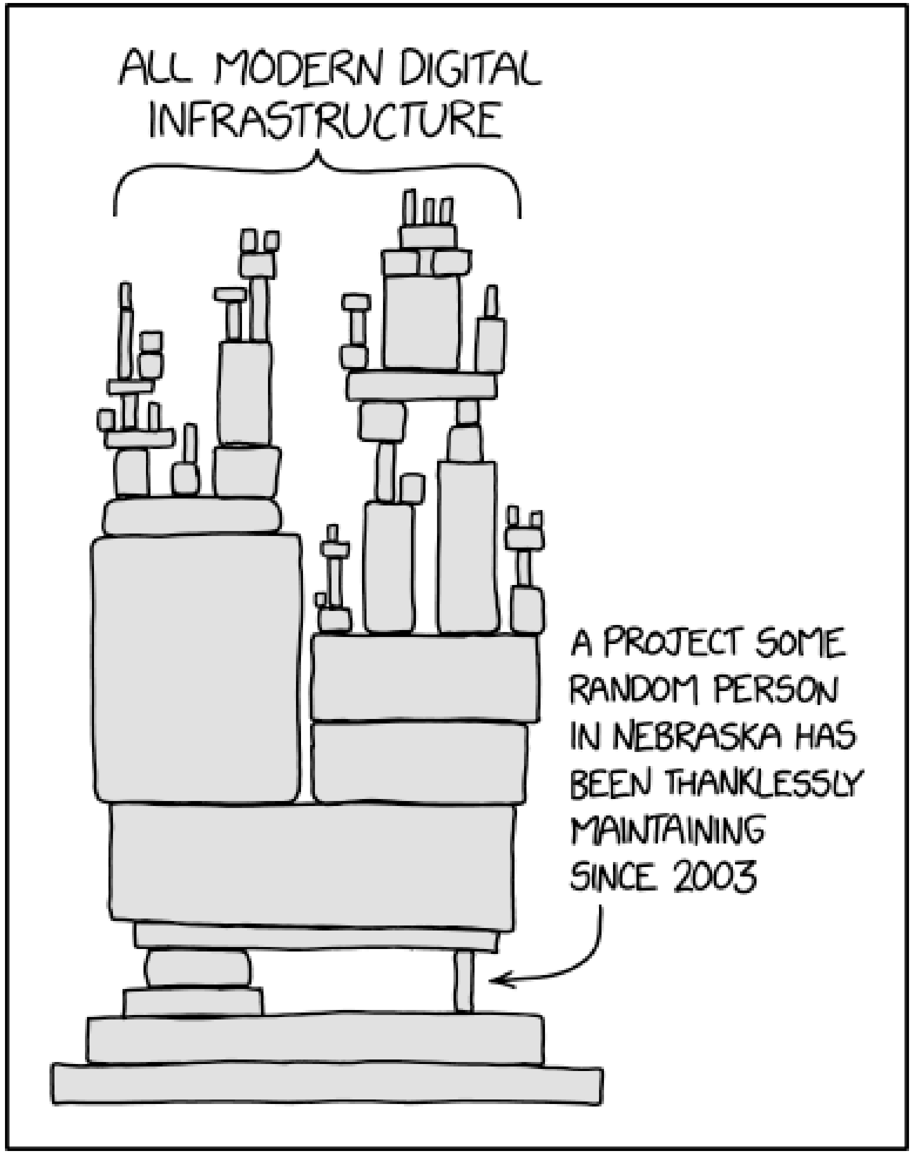
given day, etc. Most such computations are conceptually straightforward. However, the input data is usually large and the computations have to be distributed across hundreds or thousands of machines in order to finish in a reasonable amount of time. The issues of how to parallelize the computation, distribute the data, and handle failures conspire to obscure the original simple computation with large amounts of complex code to deal with these issues.

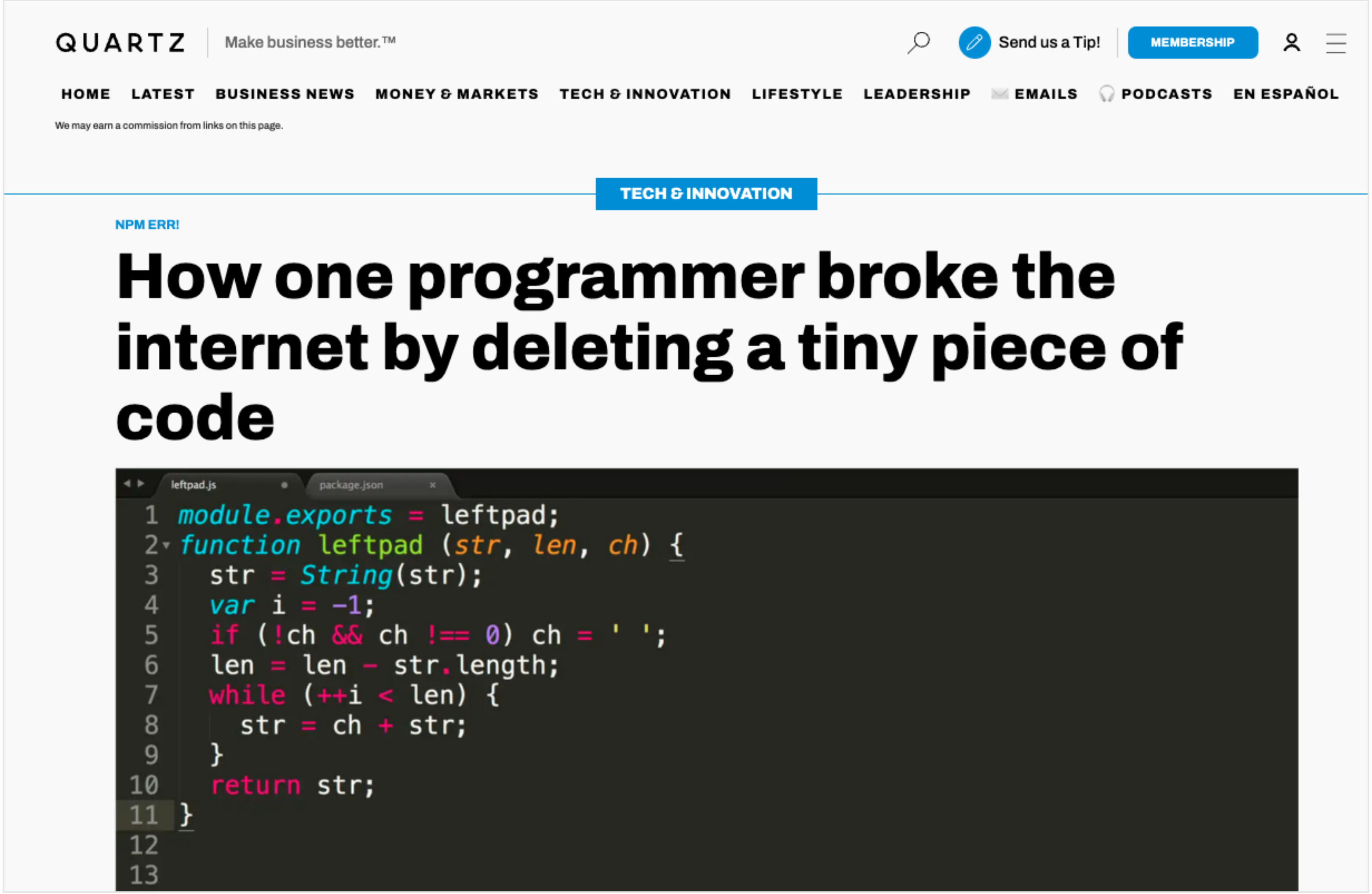
As a reaction to this complexity, we designed a new abstraction that allows us to express the simple computations we were trying to perform but hides the messy details of parallelization, fault-tolerance, data distribution and load balancing in a library. Our abstraction is inspired by the *map* and *reduce* primitives present in Lisp



Use of open source software within companies

- Is the license compatible with our intended use?
  - More on this later
- How will we handle versioning and updates?
  - Does every internal project declare its own versioned dependency or do we all agree on using one fixed (e.g., latest) version?
  - Sometimes resolved by assigning internal “owners” of a third-party dependency, who are responsible for testing updates and declaring allowable versions.
- How to handle customization of the OSS software?
  - Internal forks are useful but hard to sync with upstream changes.
  - One option: Assign an internal owner who keeps internal fork up-to-date with upstream.
  - Another option: Contribute all customizations back to upstream to maintain clean dependencies.
- Security risks? Supply chain attacks on the rise.


ALL MODERN DIGITAL INFRASTRUCTURE
A PROJECT SOME RANDOM PERSON IN NEBRASKA HAS BEEN THANKLESSLY MAINTAINING SINCE 2003



We may earn a commission from links on this page.

NPM ERR!

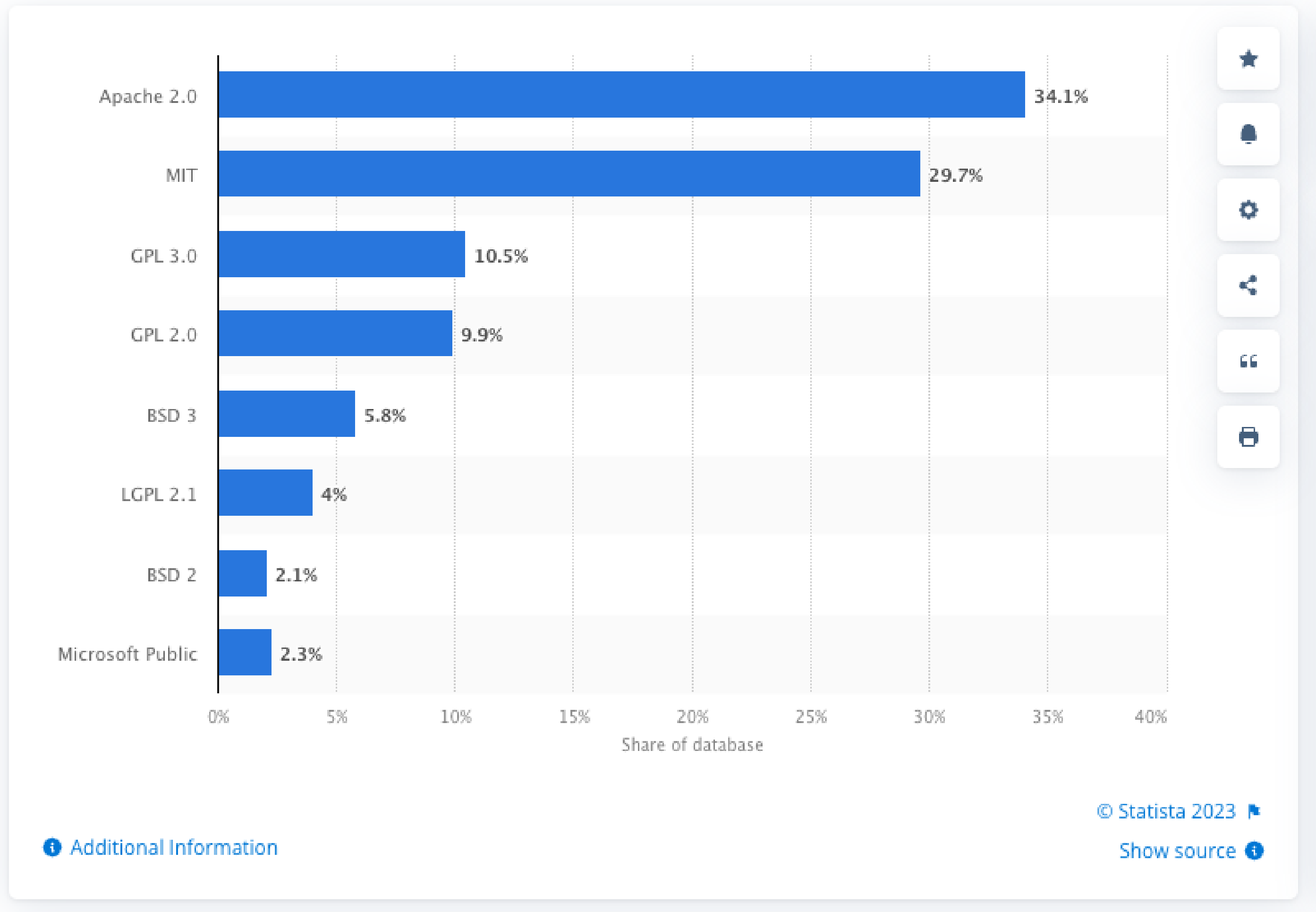
How one programmer broke the internet by deleting a tiny piece of code

```
module.exports = leftpad;
function leftpad (str, len, ch) {
  str = String(str);
  var i = -1;
  if (!ch && ch !== 0) ch = ' ';
  len = len - str.length;
  while (++i < len) {
    str = ch + str;
  }
  return str;
}
```

Software Licenses

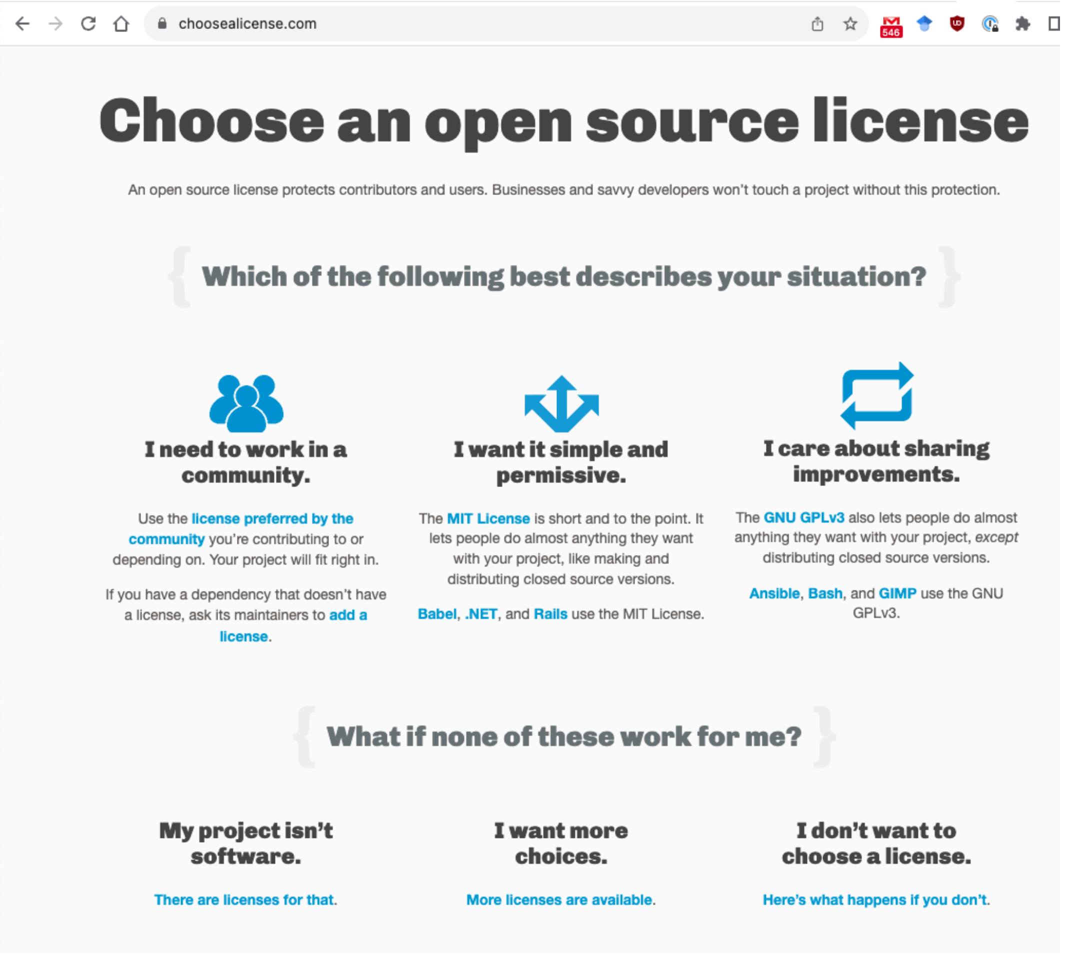
Note: I am not a lawyer (this is not legal advice)

Most popular open source licenses worldwide in 2021



https://www.statista.com/statistics/1245643/worldwide-leading-open-source-licenses/

Which license to choose?



choosealicense.com

Choose an open source license

An open source license protects contributors and users. Businesses and savvy developers won't touch a project without this protection.

Which of the following best describes your situation?

I need to work in a community.

Use the license preferred by the community you're contributing to or depending on. Your project will fit right in.

If you have a dependency that doesn't have a license, ask its maintainers to add a license.

I want it simple and permissive.

The MIT License is short and to the point. It lets people do almost anything they want with your project, like making and distributing closed source versions.

Babel, .NET, and Rails use the MIT License.

I care about sharing improvements.

The GNU GPLv3 also lets people do almost anything they want with your project, *except* distributing closed source versions.

Ansible, Bash, and GIMP use the GNU GPLv3.

What if none of these work for me?

My project isn't software.

There are licenses for that.

I want more choices.

More licenses are available.

I don't want to choose a license.

Here's what happens if you don't.

GNU General Public License: The Copyleft License

- Nobody should be restricted by the software they use. There are four freedoms that every user should have:
  - the freedom to use the software for any purpose,
  - the freedom to change the software to suit your needs,
  - the freedom to share the software with your friends and neighbors, and
  - the freedom to share the changes you make.
- Code must be made available
- Any modifications must be relicensed under the same license (copyleft)

Risks of “copyleft” licenses

- Example: GNU GPL
- Require licensing derivative works also with same license
  - This is intentional!
- Depending on a GPL project from within a proprietary or differently-licensed codebase is disaster
  - Viral effect of polluting everything else with GPL requirement
- Most companies will avoid GPL code with a ten-foot pole
  - Expect vetting process before engineers are allowed to use third-party libraries from GitHub, etc.

Lesser GNU Public License (LGPL)

- Software must be a library
- Similar to GPL but does not consider dynamic binding as “derivative work”
- So, proprietary code can depend on LGPL libraries as long as they are not being modified

- See also: GPL with classpath exception (e.g., Oracle JDK)

MIT License

- Simple, commercial-friendly license
- Must retain copyright credit
- Software is provided as is
- Authors are not liable for software
- No other restrictions

Apache License

- Similar to MIT license
- Not copyleft
- Not required to distribute source code
- Does not grant permission to use project’s trademark
- Does not require modifications to use the same license

BSD License

- No liability and provided as is.
- Copyright statement must be included in source and binary
- The copyright holder does not endorse any extensions without explicit written consent

Creative Commons (CC)

- More common for licensing data-sets instead of code
  - Examples: images, websites, documentation, slides, plots, videos

- CC-BY (attribution only; derivatives allowed)
- CC-BY-SA (attribution and share-alike for derivates)
- CC-BY-ND (attribution and no derivatives)

Dual License Business Model

- Released as GPL which requires a company using the open source product to open source it's application
- Or companies can pay $2,000 to $10,000 annually to receive a copy of MySQL with a more business friendly license

Risk: Incompatible Licenses

- Sun open-sourced OpenOffice, but when Sun was acquired by Oracle, Oracle temporarily stopped the project.
- Many of the community contributors banded together and created LibreOffice
- Oracle eventually released OpenOffice to Apache
- LibreOffice changed the project license so LibreOffice can copy changes from OpenOffice but OpenOffice cannot do the same due to license conflicts

Copyright vs. Intellectual Property (IP)

- IP and Patents cover an idea for solving a problem
  - Examples: Machine designs, pharma processes to manufacture certain drugs, (controversially) algorithms
  - Have expiry dates. IP can be licensed or sold/transferred for $$$.
- Copyrights cover particular expressions of some work
  - Examples: Books, music, art, source code
  - Automatic copyright assignment to all new work unless a license authorizes alternative uses.
- Exceptions for trivial works and ideas.

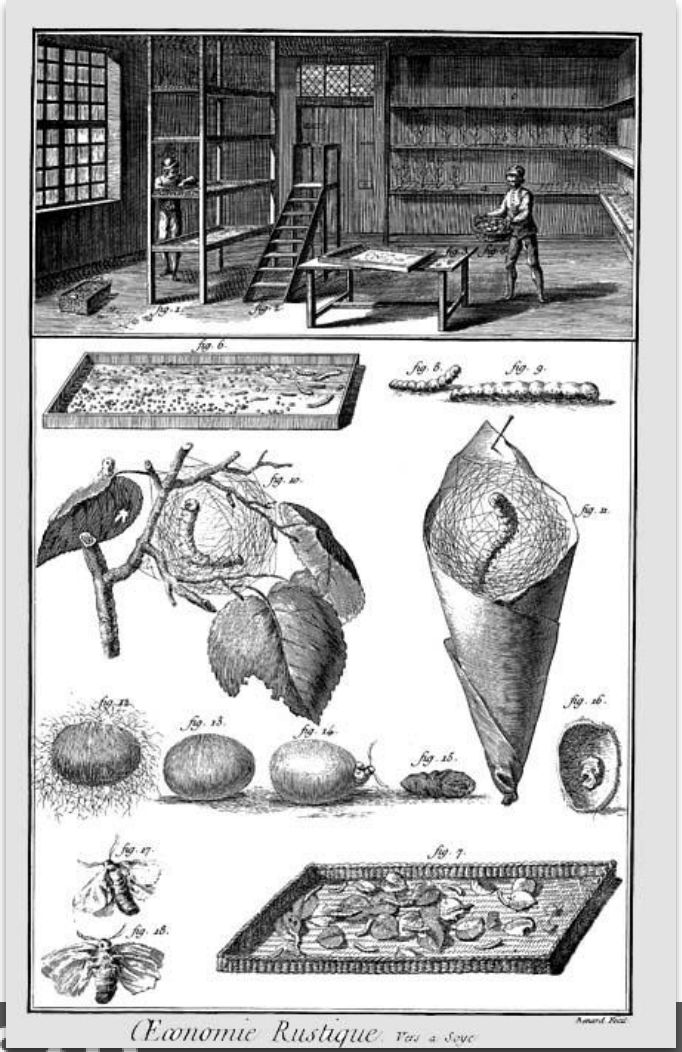
Contributor Licensing Agreements (CLA)

- Often a requirement to sign these before you can contribute to OSS projects
  - Scoped only to that project
- Assigns the maintainers specific rights over code that you contribute
  - Without this, you own the copyright and IP for even small bug fixes and that can cause them legal headaches in the future

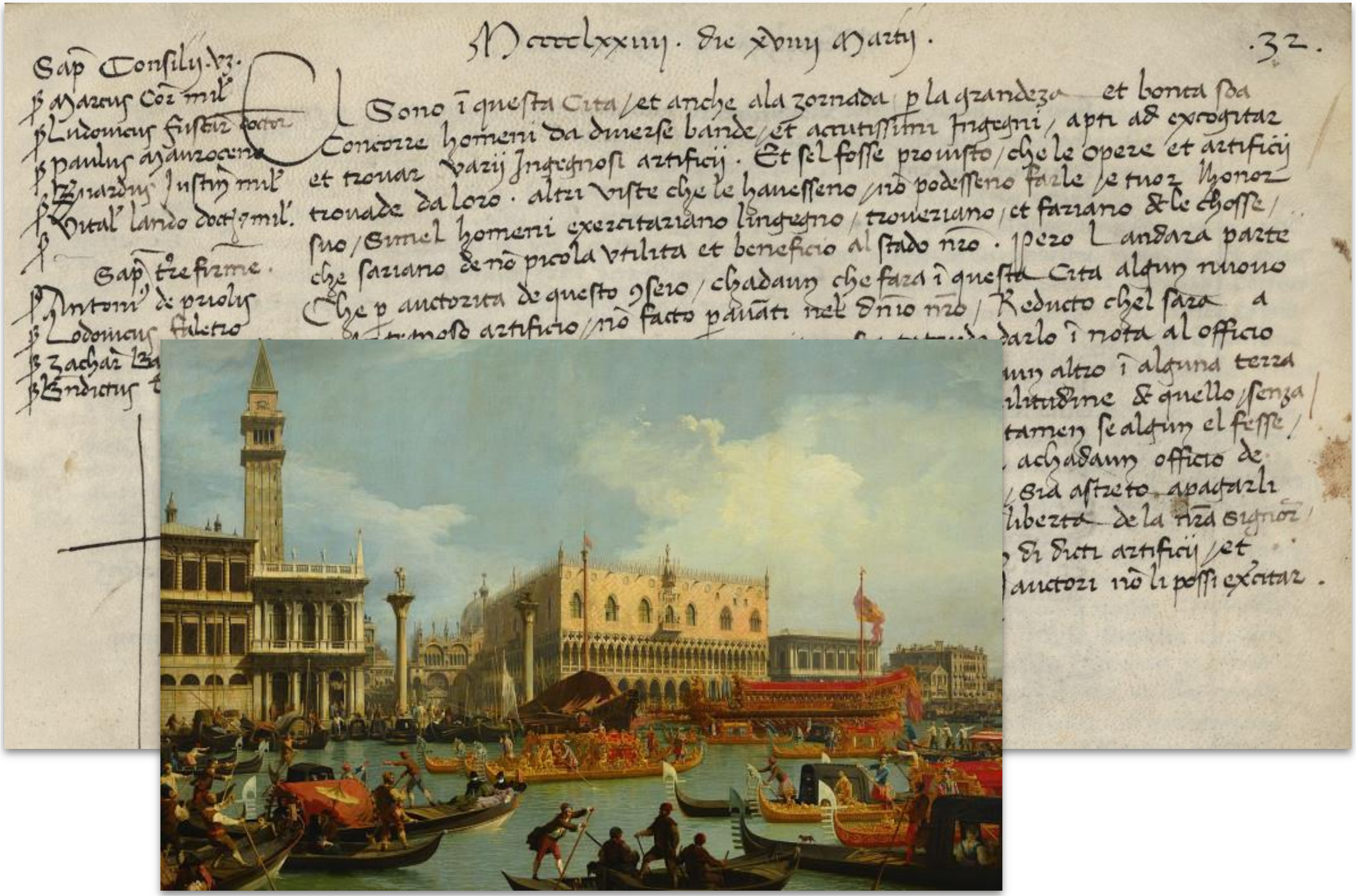
Software Patents

Software Patents:
The Good, The Bad, and The Ugly

Venice, 1474



Œconomie Rustique, Vers a Soye.



MCCCClxxiiij. die xviiij Marcij. .32.

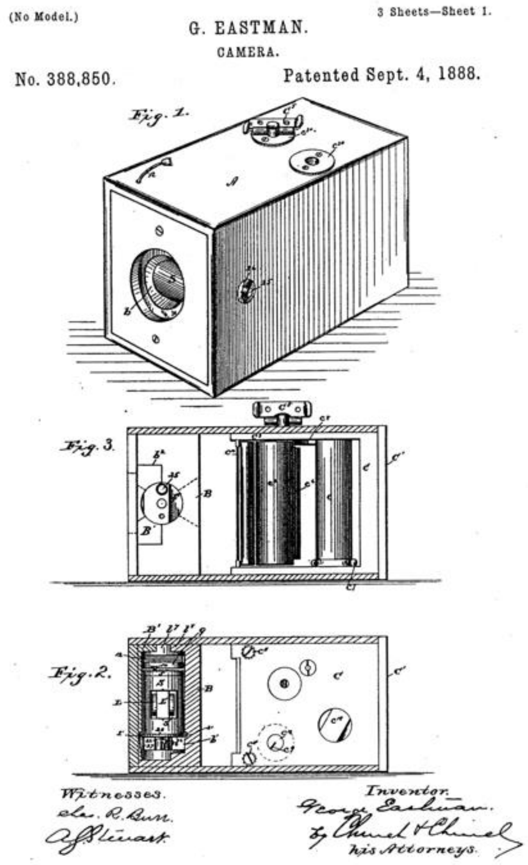
Sap Consilij. v3.
ß Marcus Cor mil
ß Ludovicus Fuscar doctor
ß paulus Maurocen
ß Girardus Iustin mil
ß Vital lando doct et mil.
ß

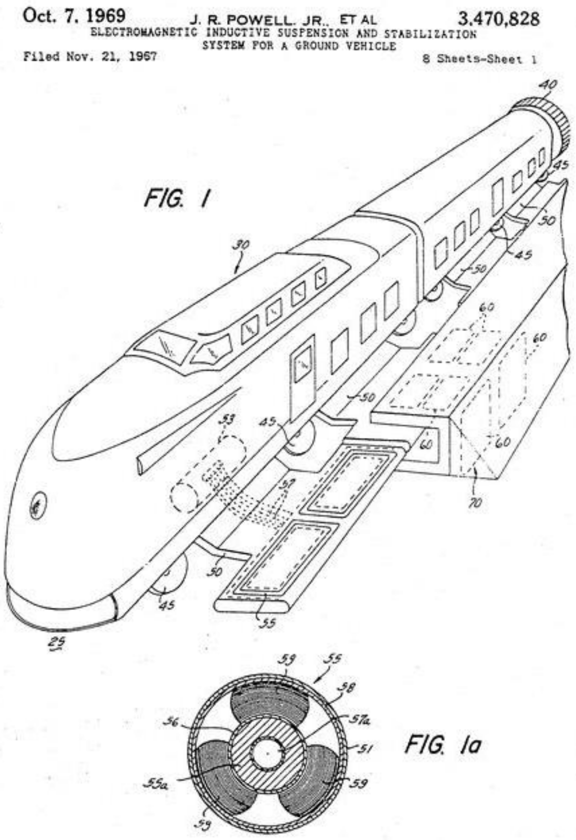
Sap terrefirme.
ß Antoni de priolis
ß Lodovicus Faletro
ß Zachar Ba
ß Benedictus

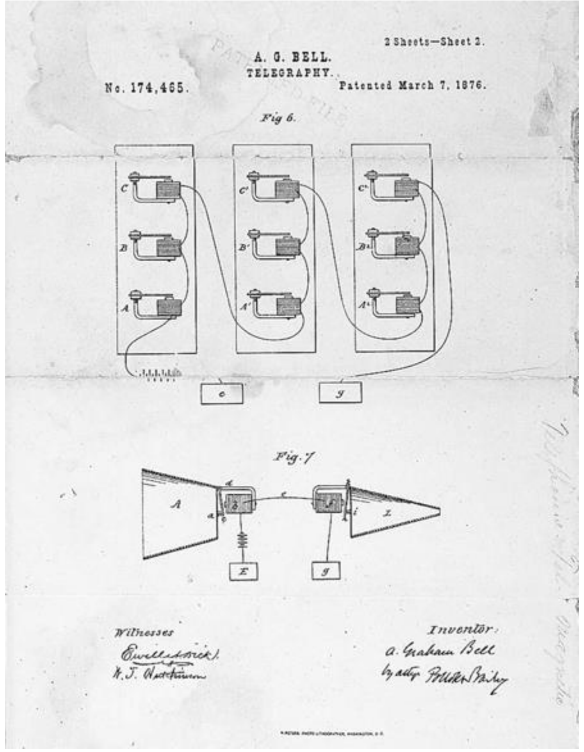
El Sono i questa Cita / et anche ala zornada p la grandeza et bonta sua
Concorre homeni da diverse bande / et acutissimi ingegni / apti ad excogitar
et trovar varij Ingegnosi artificij. Et sel fosse provisto / che le opere et artificij
trovade da loro. altri viste che le havessero / no podessero farle / e tuor lhonor
suo / Simel homeni exercitariano lingegno / troveriano / et fariano de le chosse /
che sariano de no picola utilita et beneficio al stado nro. pero landara parte
Che p auctorita de questo conseio / chadaun che fara i questa Cita algun nuovo
ingegnoso artificio / no facto pavanti nel Dnio nro / Reducto chel sara a
darlo i nota al officio
un altro i alguna terra
ilitudine de quello / senza
tamen se algun el fesse /
a chadaun officio de
/ sia astreto apagarli
liberta de la nra signor
di dicti artificij / et
auctori no li possi exercitar.

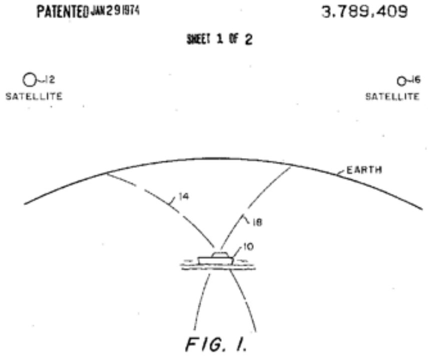
England, 1566

Today: USA









https://www.popularmechanics.com/technology/design/g20051677/patents-changed-the-world

What is a patent? New. Useful. Non-obvious.

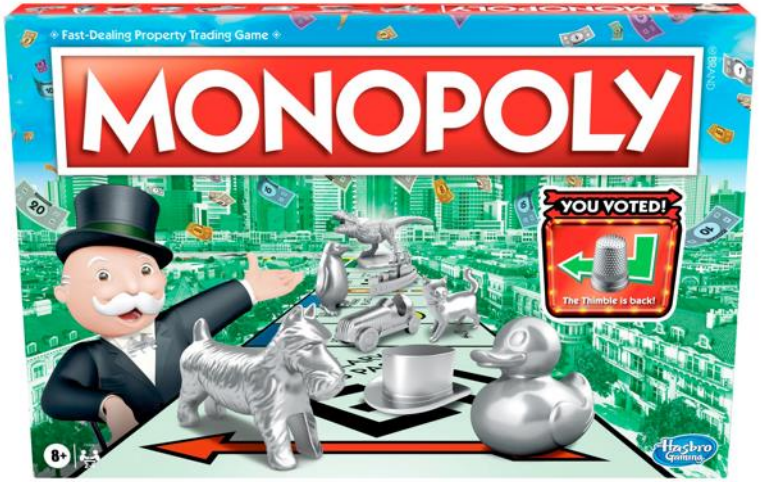
“A patent is an exclusive right granted for an invention, which is a product or a process that provides, in general, **a new way of doing something**, or offers a **new technical solution to a problem**. To get a patent, technical information about the invention must be disclosed to the public in a patent application.”

WIPO
WORLD
INTELLECTUAL PROPERTY
ORGANIZATION

https://www.wipo.int/patents/en

What rights do patents grant?

- Patents **don’t** give you the right to make, use, or sell an invention.
- Patents **do** give you the right to **exclude others** from making, using, and selling an invention for the term of a patent (20 years)
  - stop or sue others
  - licensing and royalties



What’s the difference? Patents vs. Copyright

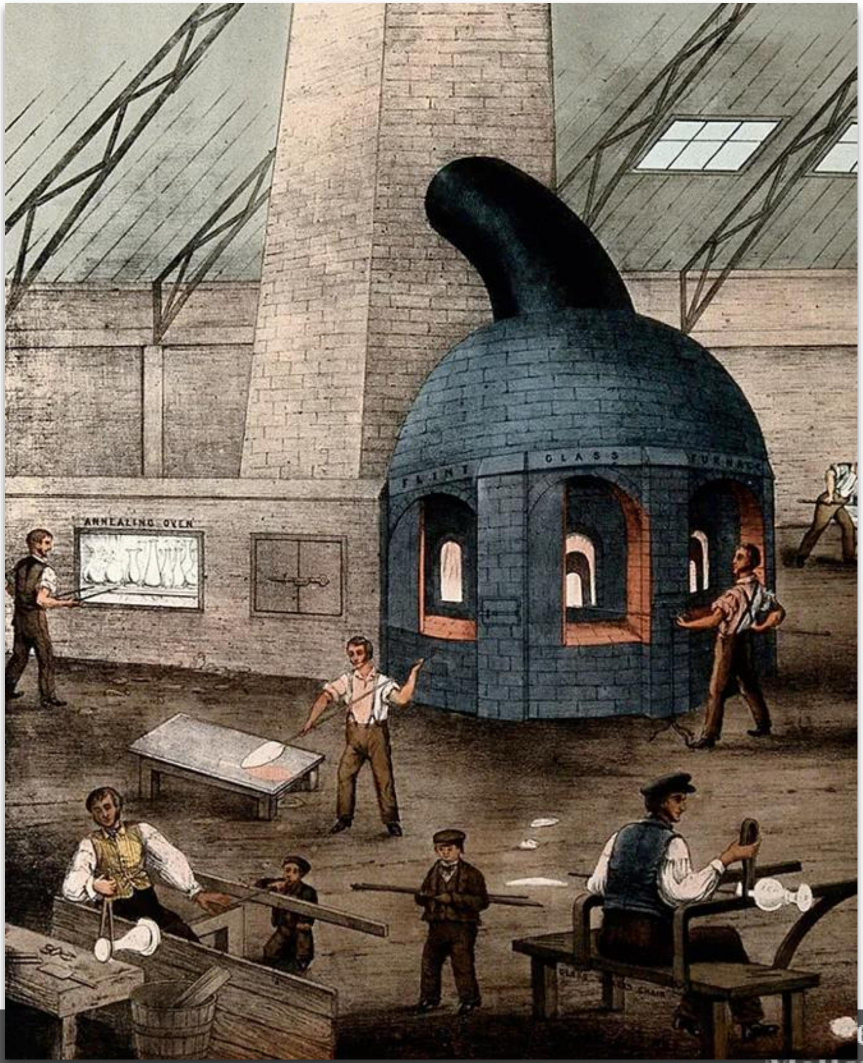
- Copyrights cover the details of expression of a work
- Copyrights don't cover any ideas
  Patents only cover ideas and the use of ideas
- Copyrights happen automatically.
  Patents are issued by a patent office in response to an application.

(No Model.)
T. A. EDISON.
LAMP BASE.
No. 438,310.
Patented Oct. 14, 1890.

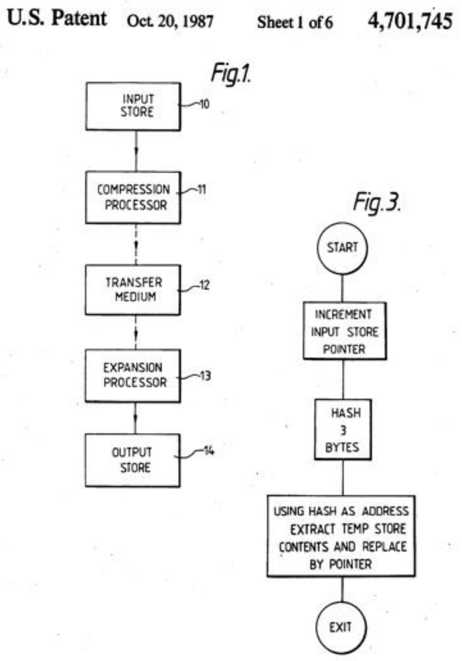
©

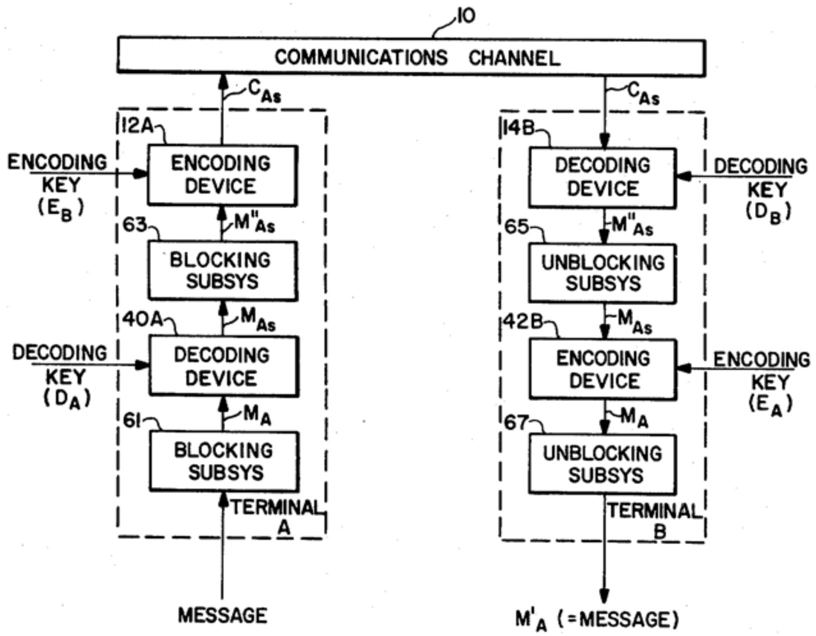
Why do patents exist?

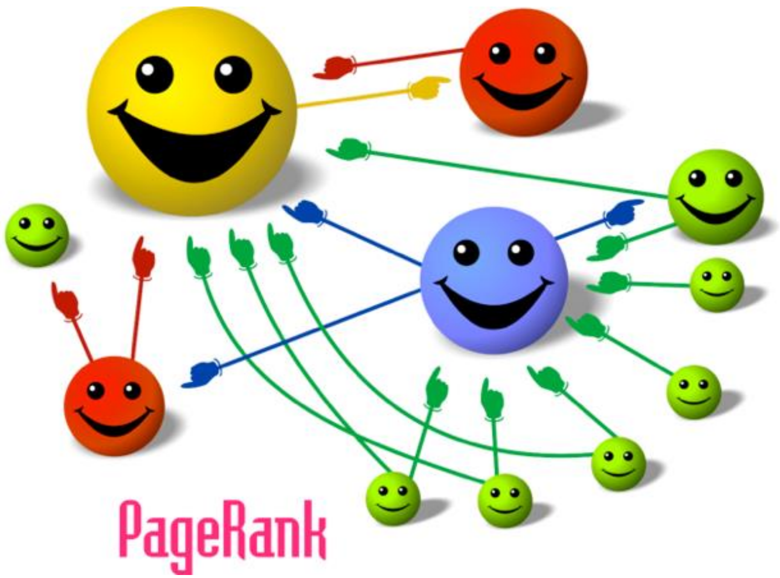
- Encourage disclosure of inventions
- Reward invention and creativity
- Protect investment of capital into R&D
- Encourage the market to “design around”
- Protect small companies from large ones



Software Patents







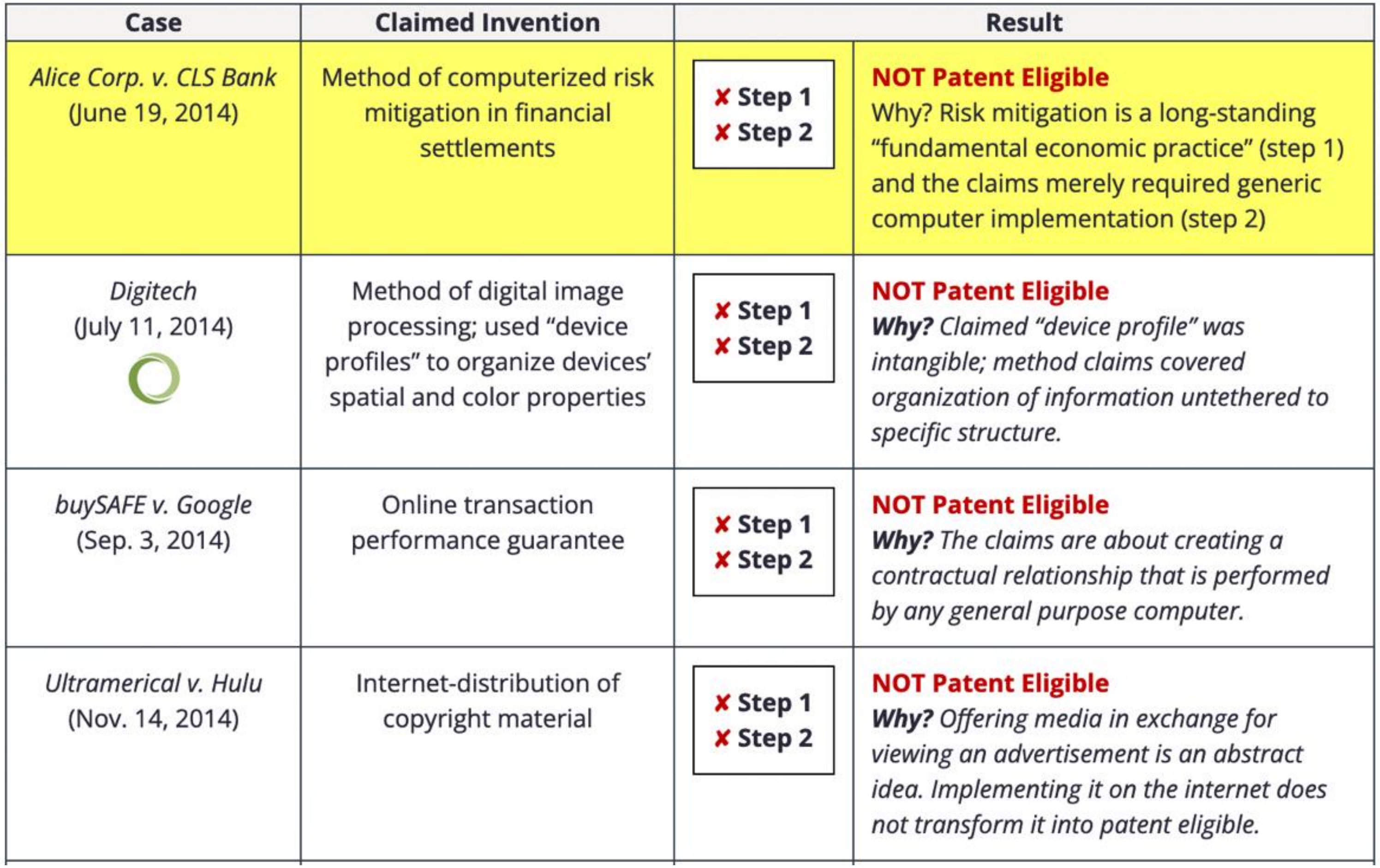
Patent or not?

Patent or not?

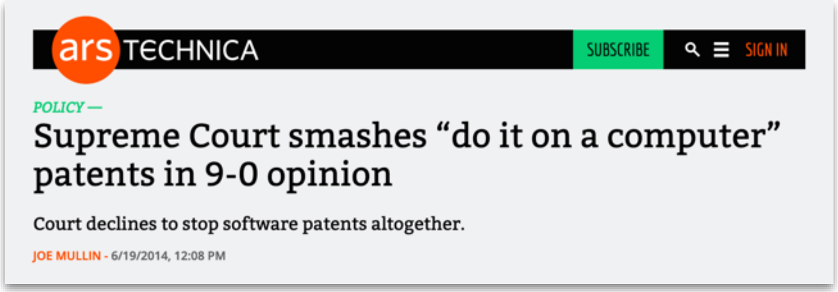
1. Running bingo on a computer
2. Using a computer to help users plan meals while achieving diet goals
3. Using a computer to order a pizza with customized toppings
4. Prompting a user before establishing a new network connection
5. Automatically notifying users when an item is picked up or delivered
6. Using a computer network to ask people to complete tasks and then wait for them to do them
7. Using SMS to perform tasks (e.g., checking bank balance)
8. Selecting ALL images in a CAPTCHA that match a given text

The software patent system is broken!

Alice vs. CLS Bank (2014)



Case	Claimed Invention	Result	
Alice Corp. v. CLS Bank (June 19, 2014)	Method of computerized risk mitigation in financial settlements	✘ Step 1 ✘ Step 2	**NOT Patent Eligible** Why? Risk mitigation is a long-standing "fundamental economic practice" (step 1) and the claims merely required generic computer implementation (step 2)
Digitech (July 11, 2014)	Method of digital image processing; used "device profiles" to organize devices' spatial and color properties	✘ Step 1 ✘ Step 2	**NOT Patent Eligible** ***Why?*** *Claimed "device profile" was intangible; method claims covered organization of information untethered to specific structure.*
buySAFE v. Google (Sep. 3, 2014)	Online transaction performance guarantee	✘ Step 1 ✘ Step 2	**NOT Patent Eligible** ***Why?*** *The claims are about creating a contractual relationship that is performed by any general purpose computer.*
Ultramerical v. Hulu (Nov. 14, 2014)	Internet-distribution of copyright material	✘ Step 1 ✘ Step 2	**NOT Patent Eligible** ***Why?*** *Offering media in exchange for viewing an advertisement is an abstract idea. Implementing it on the internet does not transform it into patent eligible.*

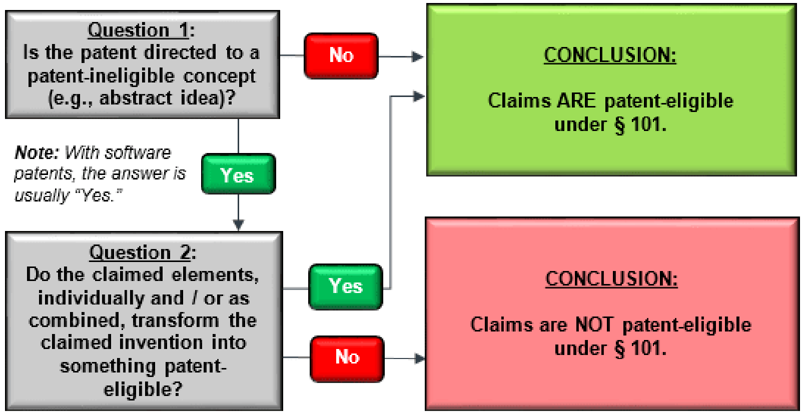


POLICY —

Supreme Court smashes "do it on a computer" patents in 9-0 opinion

Court declines to stop software patents altogether.

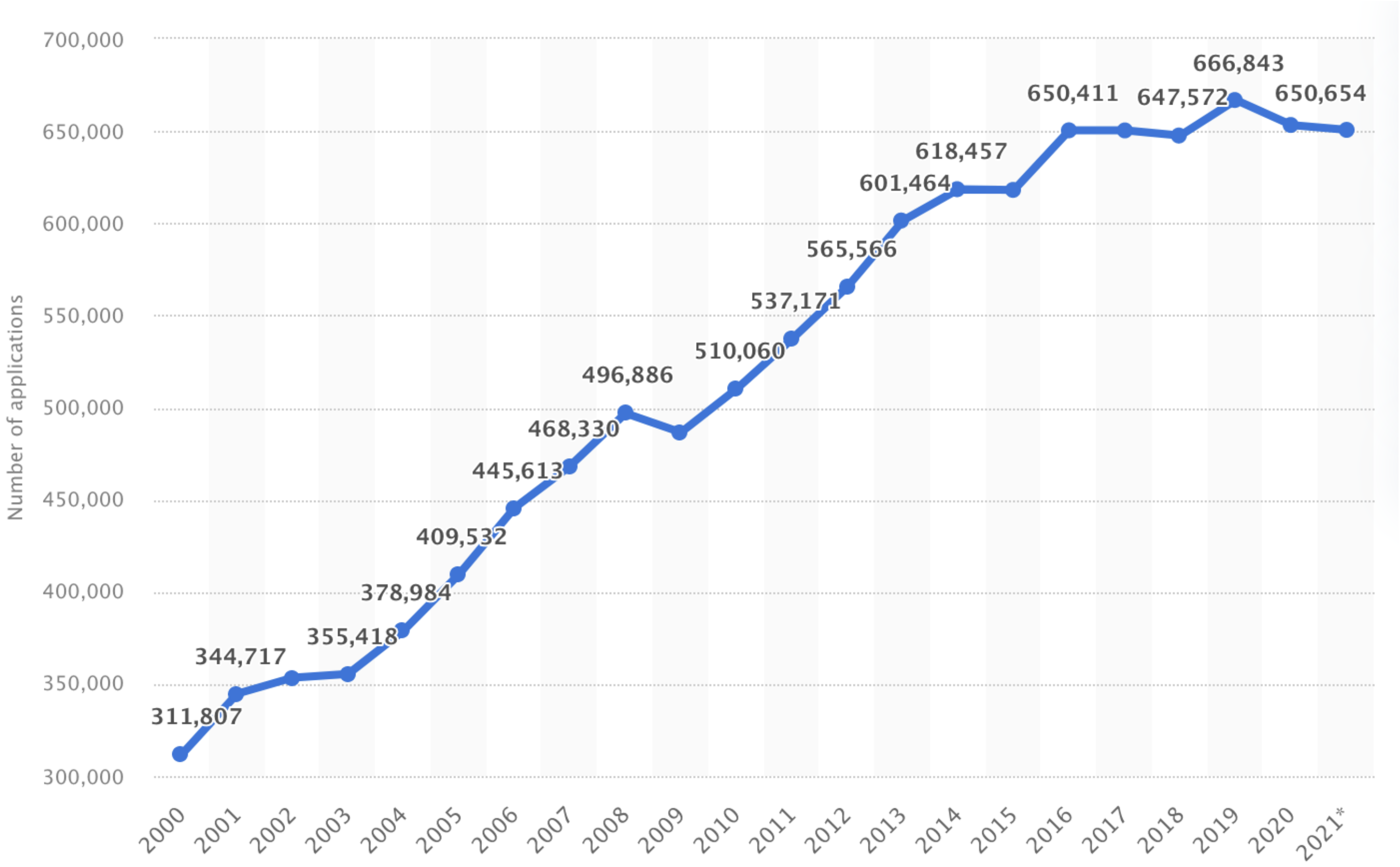
JOE MULLIN - 6/19/2014, 12:08 PM

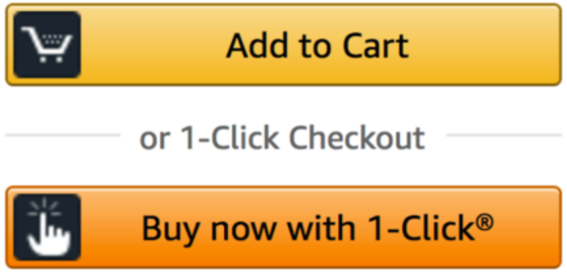


Question 1:
Is the patent directed to a patent-ineligible concept (e.g., abstract idea)?
No
Yes
Note: With software patents, the answer is usually "Yes."
Question 2:
Do the claimed elements, individually and / or as combined, transform the claimed invention into something patent-eligible?
Yes
No
CONCLUSION:
Claims ARE patent-eligible under § 101.
CONCLUSION:
Claims are NOT patent-eligible under § 101.

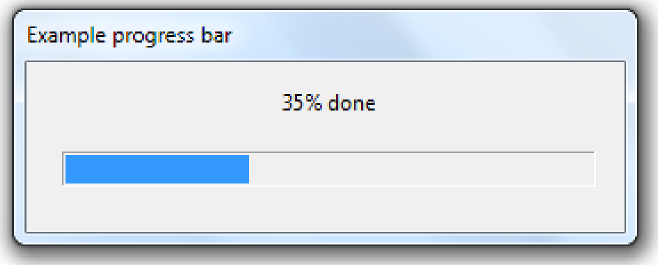
https://www.orrick.com/Articles/The-Effect-of-the-Alice-Decision-on-Software-and-3D-Printing-Patents

Problem: Inventive step and non-obviousness





US5960411A



Example progress bar

35% done

US5301348A

https://www.statista.com/statistics/256554/number-of-patent-application-filings-in-the-us

Problem: Long patent pendencies and terms

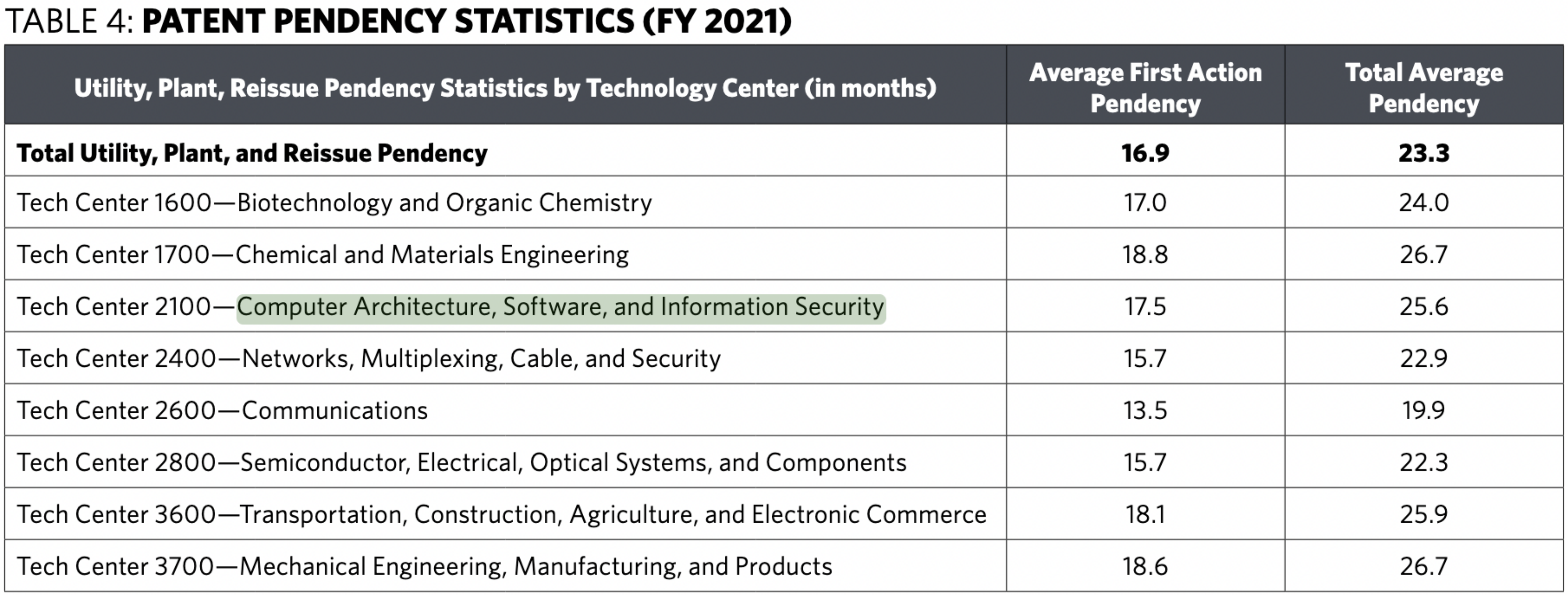


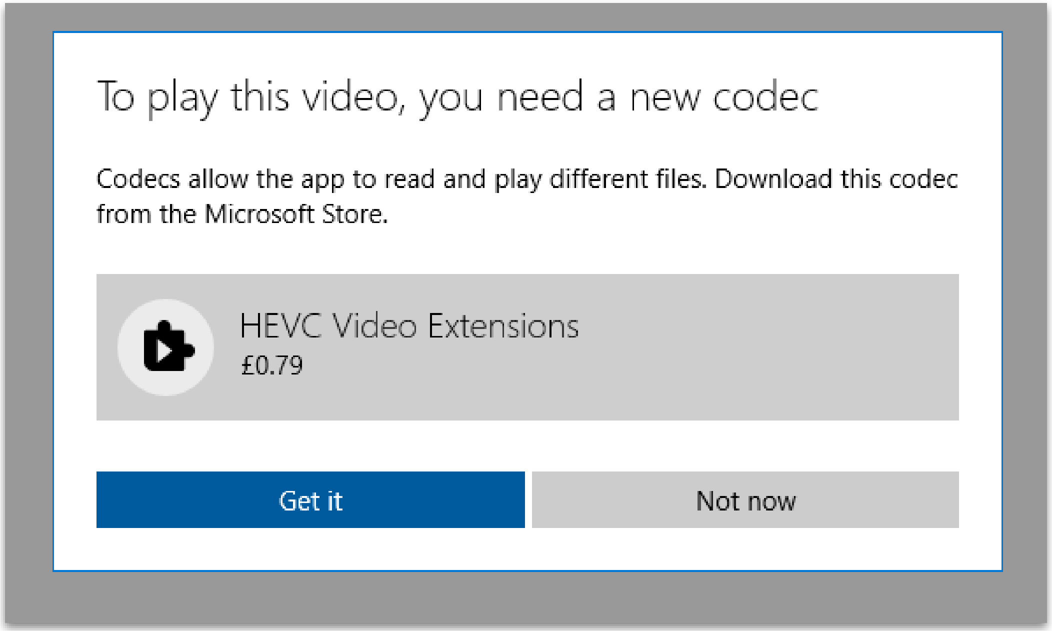
TABLE 4: **PATENT PENDENCY STATISTICS (FY 2021)**

Utility, Plant, Reissue Pendency Statistics by Technology Center (in months)	Average First Action Pendency	Total Average Pendency
Total Utility, Plant, and Reissue Pendency	**16.9**	**23.3**
Tech Center 1600—Biotechnology and Organic Chemistry	17.0	24.0
Tech Center 1700—Chemical and Materials Engineering	18.8	26.7
Tech Center 2100—Computer Architecture, Software, and Information Security	17.5	25.6
Tech Center 2400—Networks, Multiplexing, Cable, and Security	15.7	22.9
Tech Center 2600—Communications	13.5	19.9
Tech Center 2800—Semiconductor, Electrical, Optical Systems, and Components	15.7	22.3
Tech Center 3600—Transportation, Construction, Agriculture, and Electronic Commerce	18.1	25.9
Tech Center 3700—Mechanical Engineering, Manufacturing, and Products	18.6	26.7

https://www.uspto.gov/sites/default/files/documents/USPTOFY21PAR.pdf

Problem: Incompatibility

- PNG was invented to avoid GIF patent issues
- Opus is a patent-free MP3 alternative
- AV1 vs H265



To play this video, you need a new codec

Codecs allow the app to read and play different files. Download this codec from the Microsoft Store.

HEVC Video Extensions
£0.79

Problem: Independent discovery doesn’t matter!

“The idea that I can be presented with a problem, set out to logically solve it with the tools at hand, and wind up with a program that could not be legally used because someone else followed the same logical steps some years ago and filed for a patent on it is horrifying.”

John Carmack

Problem: Only large organizations benefit

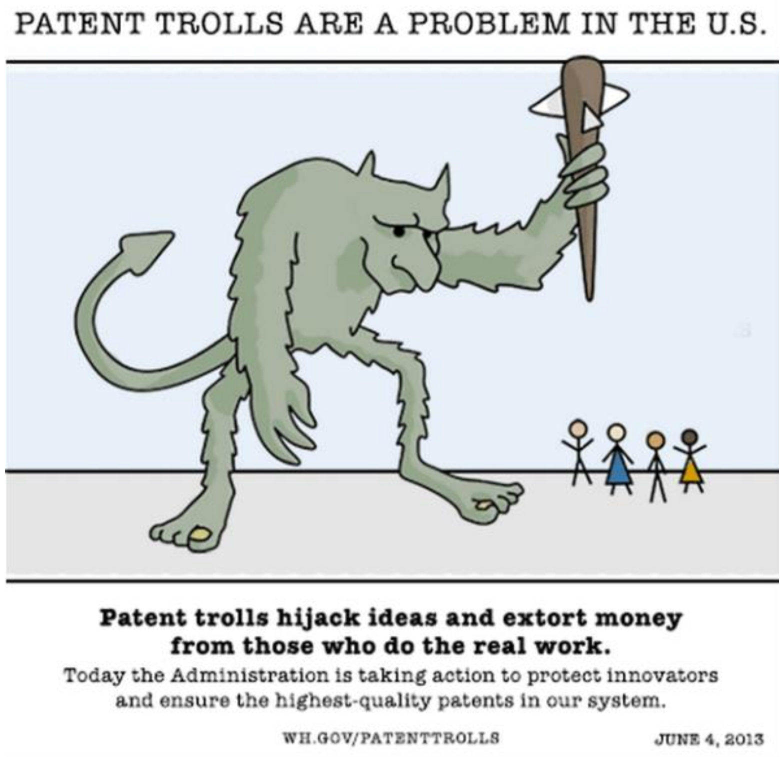
- **The patent system relies on people to challenge bad patents**
  - requires considerable time, money, and legal expertise
  - the US legal system requires both parties to pay legal fees (c.f., losers pay costs in Europe) *****
- US software patents cost between **$15,000 to $45,000!**
  - that’s before you even apply for international patents!

ELECTRONIC FRONTIER FOUNDATION'S PATENT BUSTING PROJECT

https://www.patenttrademarkblog.com/how-much-patent-costs

https://www.eff.org/issues/patent-busting-project

Problem: Non-Practicing Entities (Patent Trolls)

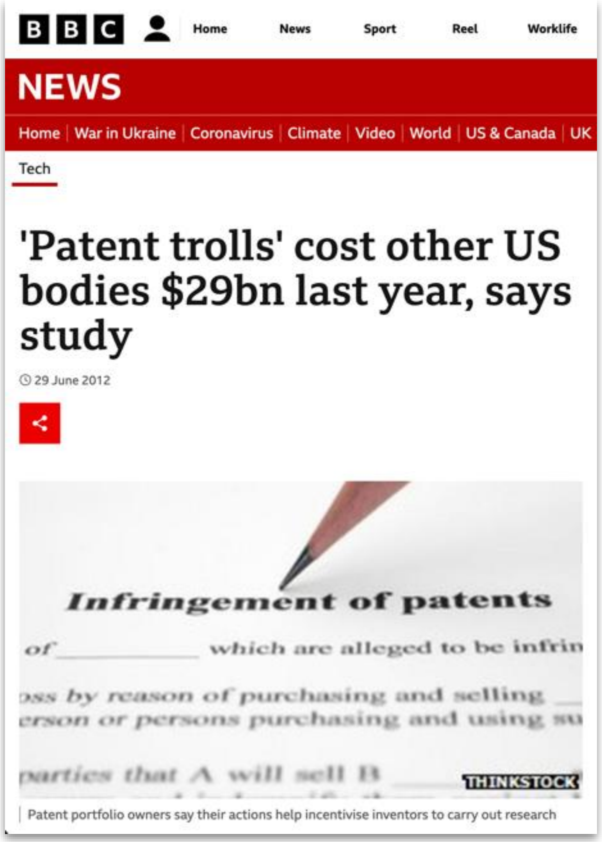


PATENT TROLLS ARE A PROBLEM IN THE U.S.

Patent trolls hijack ideas and extort money from those who do the real work.

Today the Administration is taking action to protect innovators and ensure the highest-quality patents in our system.

WH.GOV/PATENTTROLLS JUNE 4, 2013



BBC Home News Sport Reel Worklife

NEWS

Home | War in Ukraine | Coronavirus | Climate | Video | World | US & Canada | UK

Tech

'Patent trolls' cost other US bodies $29bn last year, says study

29 June 2012

Infringement of patents
of ______ which are alleged to be infrin
oss by reason of purchasing and selling
erson or persons purchasing and using su
parties that A will sell B
THINKSTOCK

Patent portfolio owners say their actions help incentivise inventors to carry out research

Problem: Innovation is Stifled

"As a developer for a small startup, absurd software patents are a constant worry. Stories abound of people like us getting pressured out of existence over the use of incredibly vague, basic interface elements and system components."

"Software patents are generally written in vague and nontechnical legal language, which obfuscates the patent in question . . . and also makes it easy to dramatically extend the patent to elements not considered at all when the patent was originally filed."

Defend Innovation

How to Fix Our Broken Patent System

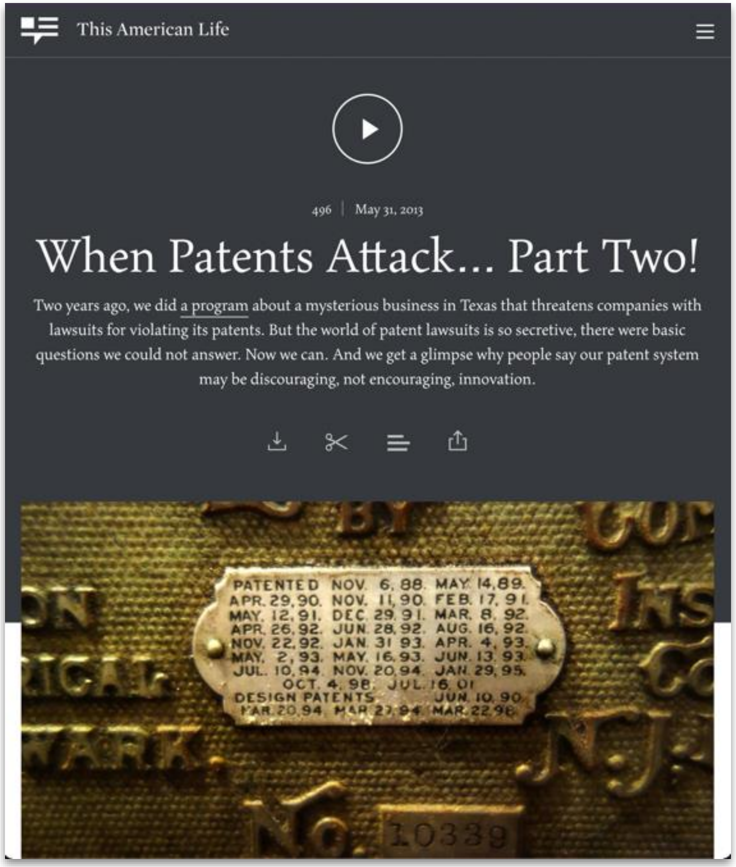
by Adi Kamdar, Daniel Nazer, Vera Ranieri

FEBRUARY 2015

ELECTRONIC FRONTIER FOUNDATION

This American Life: When Patents Attack!

- Innovatio sued libraries and coffee shops for providing WiFi in a public space
- Boadin has sued various media outlets, claiming that its patents are infringed whenever a word or phrase on your computer autocompletes
- NPHJ claims they hold a patent on “scanning and emailing documents”. They tried to sued non-profits for $1000 per employee in damages.

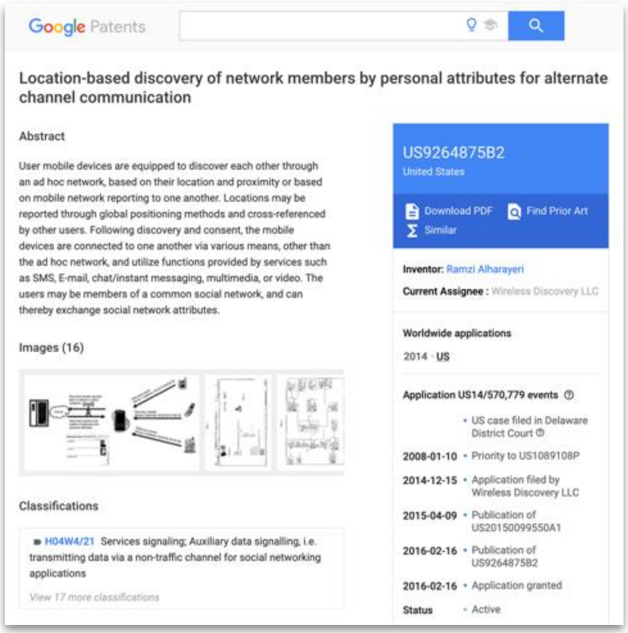


496 | May 31, 2013

When Patents Attack… Part Two!

Two years ago, we did a program about a mysterious business in Texas that threatens companies with lawsuits for violating its patents. But the world of patent lawsuits is so secretive, there were basic questions we could not answer. Now we can. And we get a glimpse why people say our patent system may be discouraging, not encouraging, innovation.

https://www.thisamericanlife.org/496/when-patents-attack-part-two



Location-based discovery of network members by personal attributes for alternate channel communication

Abstract

User mobile devices are equipped to discover each other through an ad hoc network, based on their location and proximity or based on mobile network reporting to one another. Locations may be reported through global positioning methods and cross-referenced by other users. Following discovery and consent, the mobile devices are connected to one another via various means, other than the ad hoc network, and utilize functions provided by services such as SMS, E-mail, chat/instant messaging, multimedia, or video. The users may be members of a common social network, and can thereby exchange social network attributes.

Images (16)

Classifications

H04W4/21 Services signaling; Auxiliary data signalling, i.e. transmitting data via a non-traffic channel for social networking applications

View 17 more classifications

US9264875B2
United States

Inventor: Ramzi Alharayeri

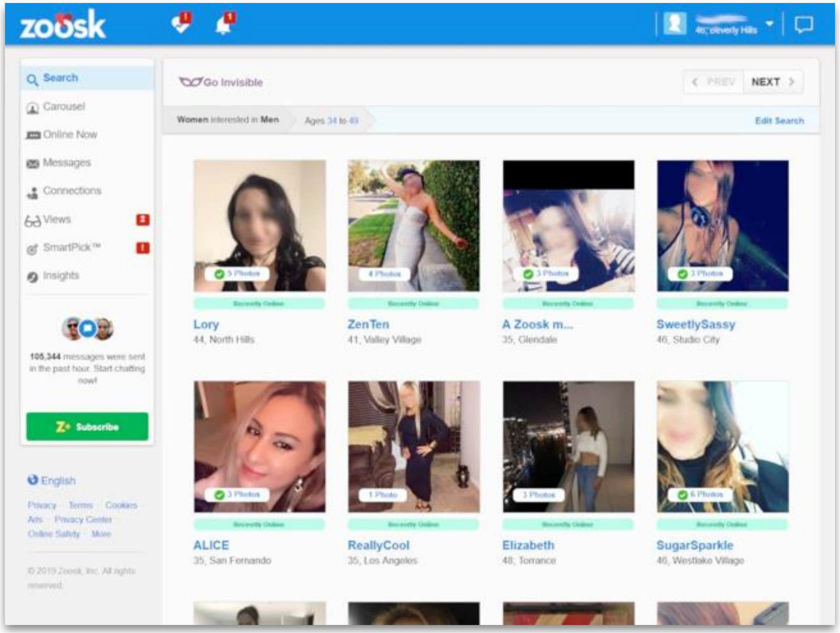
Current Assignee : Wireless Discovery LLC

Worldwide applications

2014 · US

Application US14/570,779 events

	US case filed in Delaware District Court
2008-01-10	Priority to US1089108P
2014-12-15	Application filed by Wireless Discovery LLC
2015-04-09	Publication of US20150099550A1
2016-02-16	Publication of US9264875B2
2016-02-16	Application granted
Status	Active



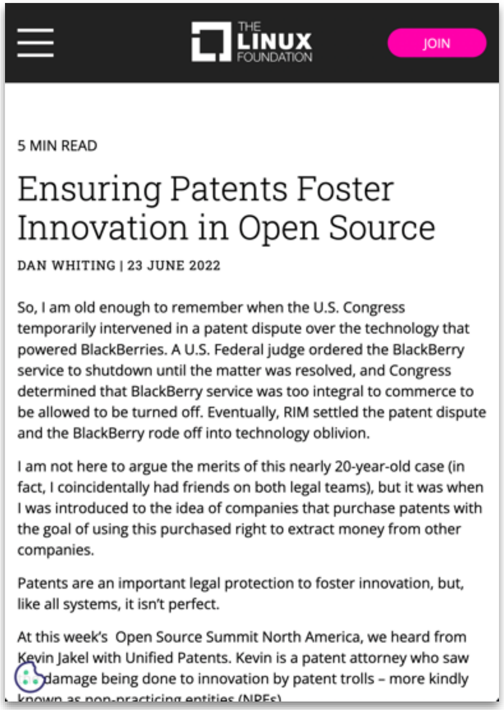
105,344 messages were sent in the past hour. Start chatting now!

ELECTRONIC FRONTIER FOUNDATION

- Zoosk has a website that mobile devices can connect to
- Zoosk's server collects information from the mobile devices, including location and unique device identifiers
- Zoosk users can send and accept invitations to connect with and send messages to each other.
- Zoosk shares profile information of connected users, who are "members of a same social network" (i.e., they're on Zoosk)
- Zoosk can connect users who are in the immediate vicinity of each other, or a particular distance away

https://www.eff.org/deeplinks/2022/05/patent-troll-uses-ridiculous-people-finder-patent-sue-small-dating-companies

Problem: Open Source is under attack, too!



5 MIN READ

Ensuring Patents Foster Innovation in Open Source

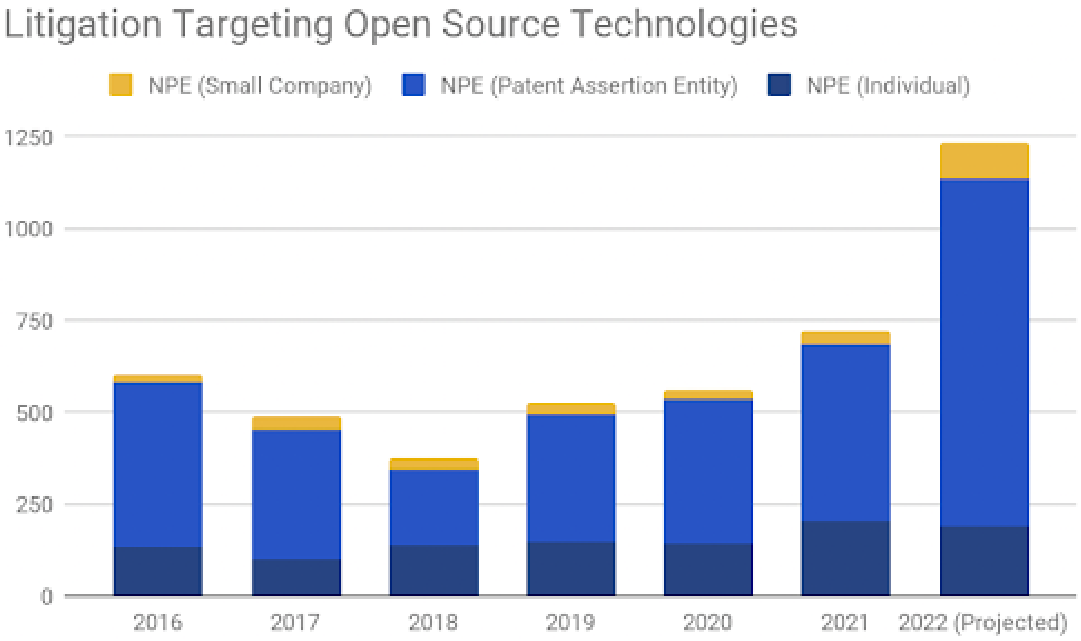
DAN WHITING | 23 JUNE 2022

So, I am old enough to remember when the U.S. Congress temporarily intervened in a patent dispute over the technology that powered BlackBerries. A U.S. Federal judge ordered the BlackBerry service to shutdown until the matter was resolved, and Congress determined that BlackBerry service was too integral to commerce to be allowed to be turned off. Eventually, RIM settled the patent dispute and the BlackBerry rode off into technology oblivion.

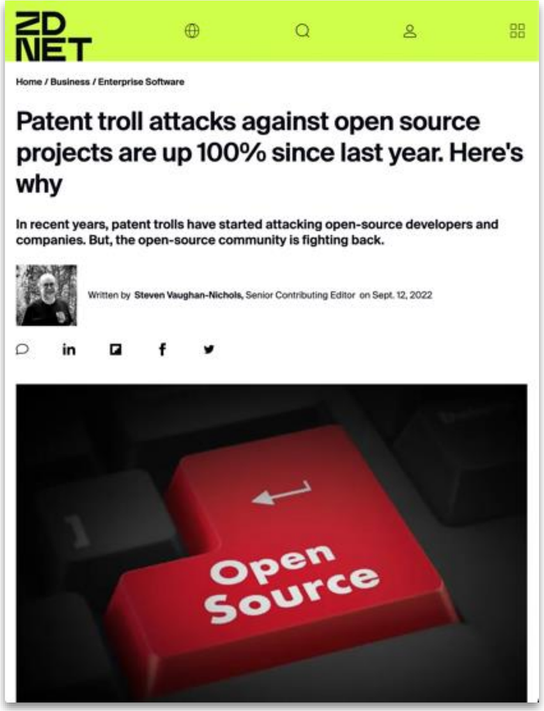
I am not here to argue the merits of this nearly 20-year-old case (in fact, I coincidentally had friends on both legal teams), but it was when I was introduced to the idea of companies that purchase patents with the goal of using this purchased right to extract money from other companies.

Patents are an important legal protection to foster innovation, but, like all systems, it isn't perfect.

At this week's Open Source Summit North America, we heard from Kevin Jakel with Unified Patents. Kevin is a patent attorney who saw damage being done to innovation by patent trolls – more kindly known as non-practicing entities (NPEs).



Data collected through June 6, 2022



Home / Business / Enterprise Software

Patent troll attacks against open source projects are up 100% since last year. Here's why

In recent years, patent trolls have started attacking open-source developers and companies. But, the open-source community is fighting back.

Written by Steven Vaughan-Nichols, Senior Contributing Editor on Sept. 12, 2022

What next?

- Alternative licensing models
  - The Defensive Patent License (DPL)
  - The Open Invention Network (OIN)
  - License on Transfer (LOT)
- Bogus patent bounties
- [Unified Patents]
- Commonsense reform
- **Abolish software patents?**

Project Jengo Redux: Cloudflare's Prior Art Search Bounty Returns

04/26/2021

Doug Kramer