Open Source

17-313 Fall 2024

Foundations of Software Engineering

https://cmu-17313q.github.io

Eduardo Feo Flushing

Learning goals

- Understand the terminology “free software” and explain open source culture and principles.
- Understand the difference between free software and open source movements
- Express an educated opinion on the philosophical/political debate between free software, open source, and proprietary principles.
- Reason about the tradeoffs of the open-source model on issues like quality and risk, both in general and in a proprietary context.

Outline

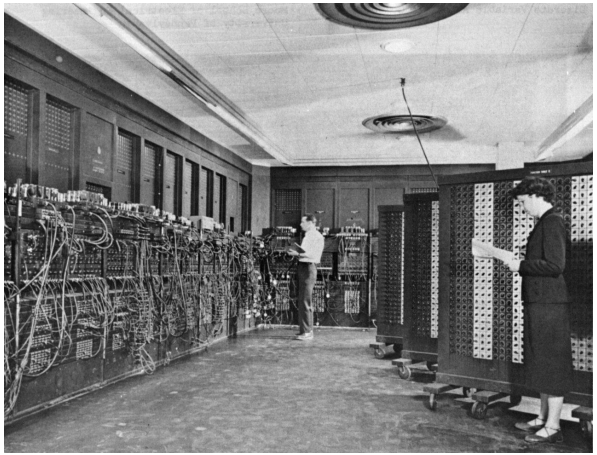
- A bit of history
- Free Software vs Open Source
  - Debate
- Open Source in the 21st Century
- Licenses

Motivation to understand open source.

- Companies work on open source projects.
- Companies use open source projects.
- Companies are based around open source projects.
- Principles percolate throughout industry.
- Political/philosophical debate, and being informed is healthy.

A bit of history: 1950s

- Grace Hopper writes the world's first compiler A-0
- IBM introduces FORTRAN
- Software released as public domain

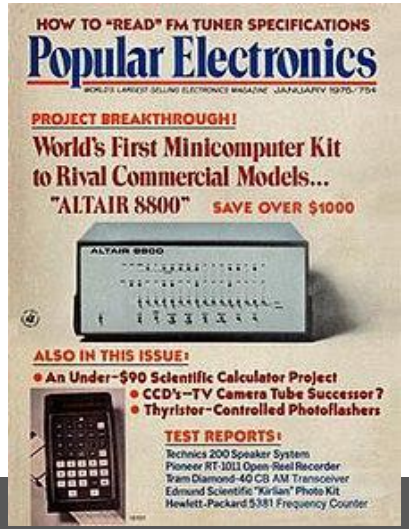


A bit of history: 1960s

- IBM's System/360 (general-purpose computing)
- The rise of "minicomputers"
  - DEC PDP-8
- "IBM and the seven dwarfs"
- Software usually bundled with hardware
- Software development is very expensive
- 1969: **"Software unbundling"**
  - **Software Industry is born**

A bit of history: 1970s

- Rise of the “microcomputers”
  - PDP, Altair
- Unix first release by Bell Labs
- Microsoft was founded
- Gate’s “Open Letter to Hobbyists”



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?

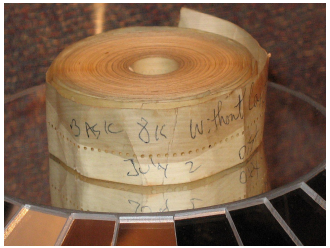
Is this fair? One thing you don't do by stealing software is get back at MITS for some problem you may have had. MITS doesn't make money selling software. The royalty paid to us, the manual, the tape and the overhead make it a break-even operation. One thing you do do is prevent good software from being 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

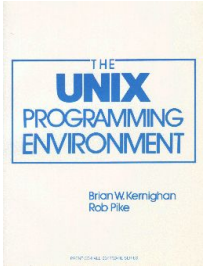
Bill Gates
General Partner, Micro-Soft

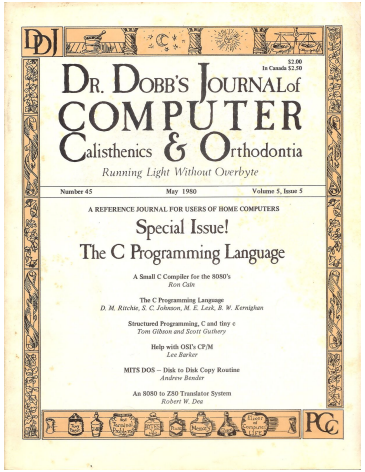


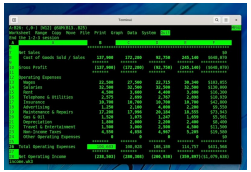
```
;*****************************************
;
;     TINY BASIC FOR INTEL 8080
;           VERSION 2.0
;         BY LI-CHEN WANG
;       MODIFIED AND TRANSLATED
;          TO INTEL MNEMONICS
;          BY ROGER RAUSKOLB
;           10 OCTOBER 1976
;             @COPYLEFT
;        ALL WRONGS RESERVED
;
;*****************************************
```

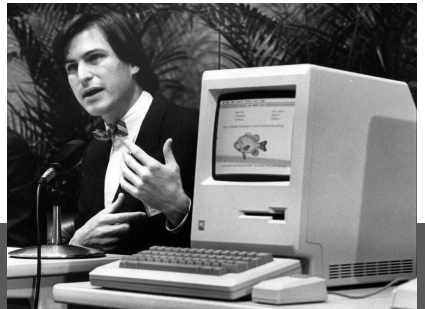
A bit of history: 1980s

- Rise “personal computers”
- Software industry flourishes
  - Adobe, Autodesk, Lotus, …
- Informal software sharing
  - Software “hobbyists”
- Apple Computer, Inc. v. Franklin Computer Corp.
- The *Unix Programming Environment* is published
- Stallman starts the **Free Software Movement**









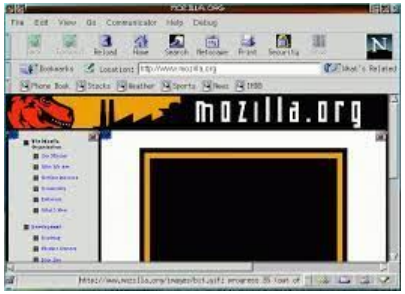
“Free as in free speech.”

GNU/Linux

FREE SOFTWARE FOUNDATION

A bit of history: 1990s

- Linux
- Open Source Initiative
- Netscape started the open source Mozilla project.



Open Source Initiative

Anchor

Do you want to support Open Source? Book a meeting with us now.

Innovator

DataStax

Premier

Bloomberg Engineering, cisco, indeed, Red Hat, Google

Maintainer

CapitalOne, COMCAST, FerretDB, GitHub, Microsoft, OpenLogic by Perforce, salesforce, slim.ai, TIDELIFT

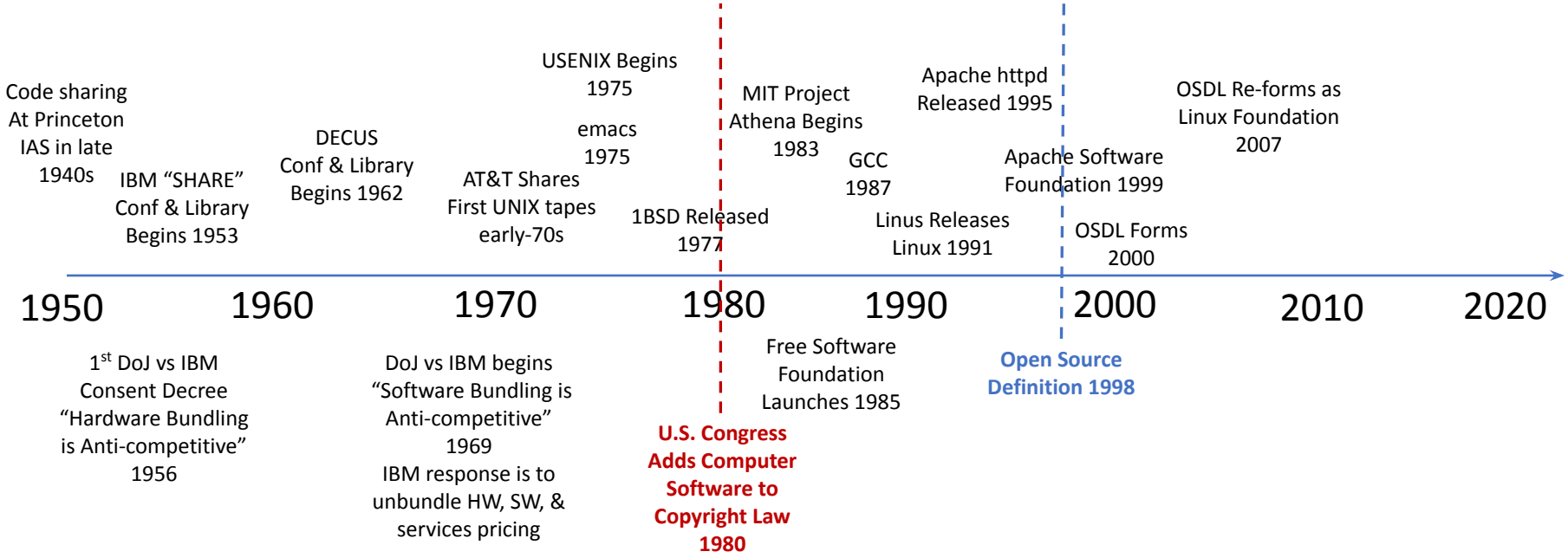
Supporter

eye/o, harness, ENGINEERING

open source initiative®

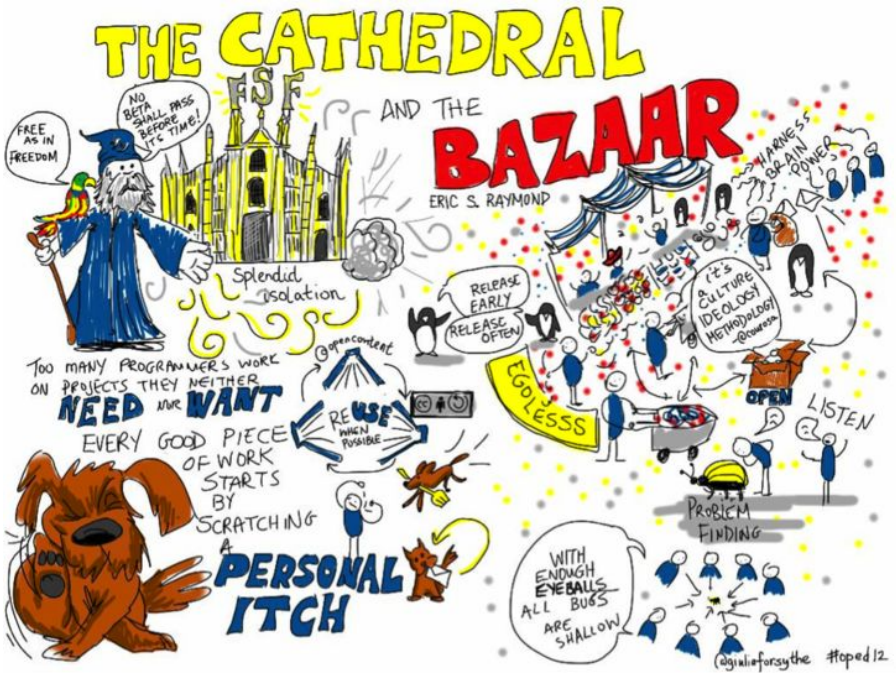
We've collaborated on software since we've created software

(Because creating good software is hard work)



Credit: Stephen Walli

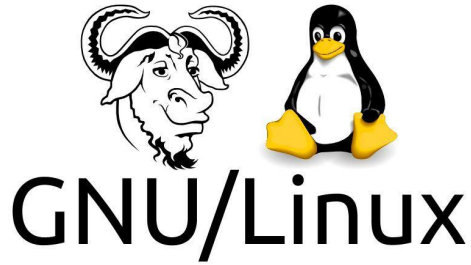
Open Source: The bazaar model



Visual Notes on Eric S. Raymond's "The Cathedral and the Bazaar." by Giulia Forsythe

Free Software vs Open Source

- Free software origins (70-80s ~Stallman)
  - Political goal
  - Software part of free speech
    - free exchange, free modification
    - proprietary software is unethical
    - security, trust
  - GNU project, Linux, GPL license



- Open source (1998 ~ OSI)
  - Rebranding without political legacy
  - Emphasis on internet and large dev./user involvement
  - Openness toward proprietary software/coexist
  - (Think: Netscape becoming Mozilla)

Definitions

freedom 0 : The freedom to run the program as you wish, for any purpose

freedom 1: The freedom to study how the program works, and change it so it does your computing as you wish

freedom 2: The freedom to redistribute copies so you can help others

freedom 3: The freedom to distribute copies of your modified versions to others

1. Free Redistribution

2. Source Code

3. Derived Works

4. Integrity of The Author's Source Code

5. No Discrimination Against Persons or Groups

6. No Discrimination Against Fields of Endeavor

7. Distribution of License

8. License Must Not Be Specific to a Product

9. License Must Not Restrict Other Software

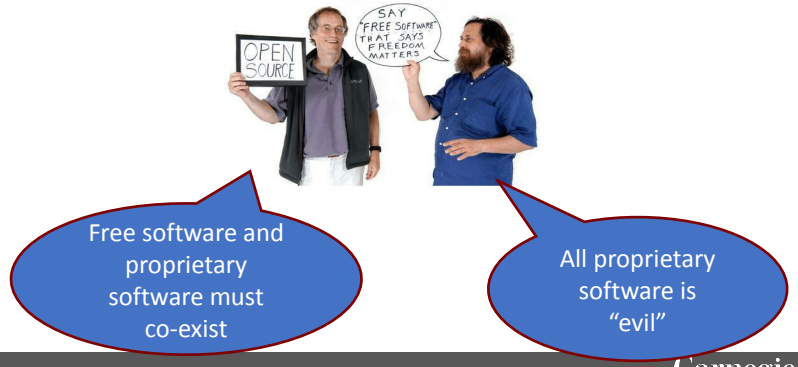
10. License Must Be Technology-Neutral

Participation Activity: Debate (Part 1)

- First rows: Advocates for Open Source
- Last rows: Advocates for Free Software

Discuss in groups and prepares three arguments in favor of your stance. Write it on a piece of paper.

Share with the class.



Participation Activity: Debate (Part 2)

- First rows: Advocates for Open Source
- Last rows: Advocates for Free Software

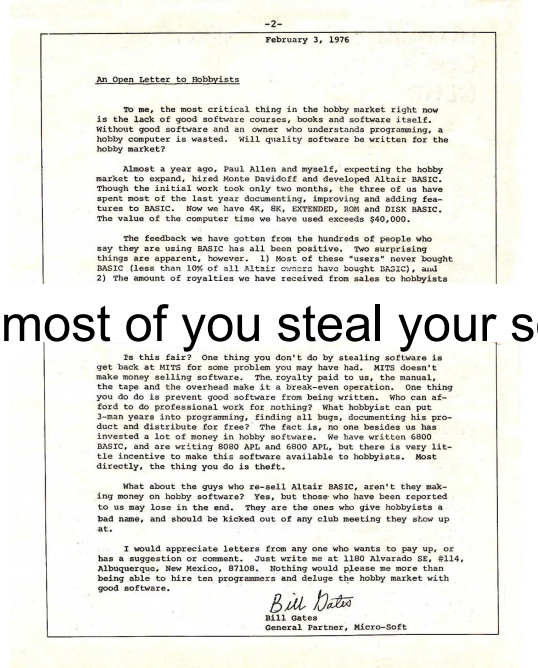
Listen carefully to the opposing team's arguments and prepare a rebuttal. The goal is to challenge the validity or practicality of one of the other group's points

Share with the class.

Perception:

- Anarchy
- Demagoguery
- Ideology
- Altruism
- Many eyes

Ecosystem has Changed



-2-

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

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

"…most of you steal your software…"

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

Wait, Open Source won?

WIRED BACKCHANNEL BUSINESS CULTURE GEAR IDEAS SCIENCE SECURITY

KLINT FINLEY BUSINESS AUG 11, 2016 7:00 AM

Open Source Won. So, Now What?

Open source now runs the world. But it still faces problems

If We've Won, Why Are We Still Explaining Open Source?

Stephen Walli
Principal Program Manager, Azure Office of the CTO at Microsoft | Adjunct Faculty, Carnegie Mellon University
+ Follow

Published Jun 19, 2015

Open Source in the 21st Century

- The Old Debate: Proprietary vs. Open Source
  - Historically, proprietary software was seen as the "safe" option, offering support and polished solutions.
  - Open Source was often viewed skeptically: “Free, but risky,” with concerns over quality and reliability.
- The Shift: Open Source Goes Mainstream
  - By the 2010s, Open Source had proven its reliability, flexibility, and cost-effectiveness.
  - Now, Open Source powers critical technologies, including:
    - 90% of the cloud (e.g., Kubernetes, Docker).
    - The backbone of AI/ML frameworks (e.g., TensorFlow, PyTorch).
    - Web development (e.g., React, Node.js)

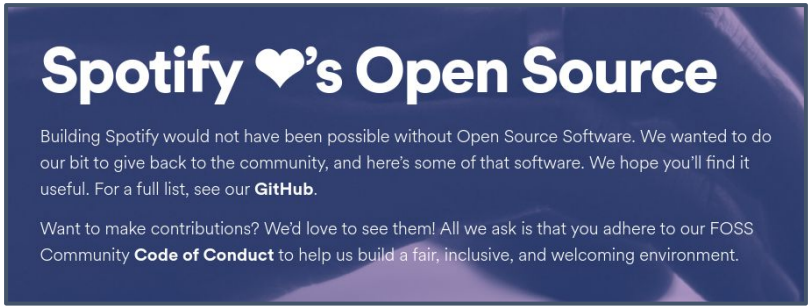
How Companies Embrace Open Source

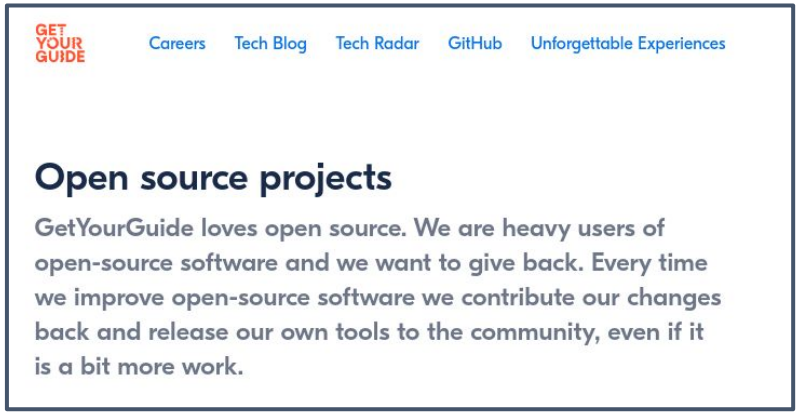
Adoption: Companies integrate Open Source into their tech stacks to save development time and cost.

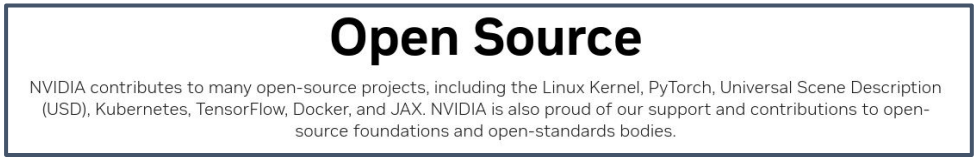
Collaboration: Firms like Google, Meta, and Red Hat actively contribute to major projects (e.g., Linux, Kubernetes).

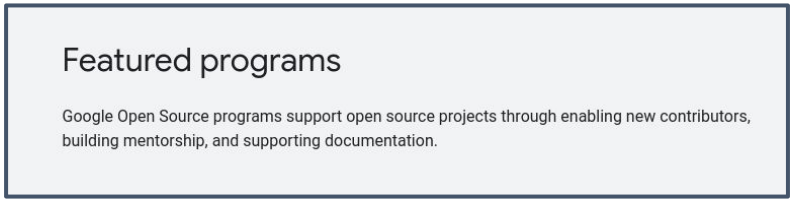
Leadership: Open Source communities are now led by industry giants, blending innovation with enterprise reliability.

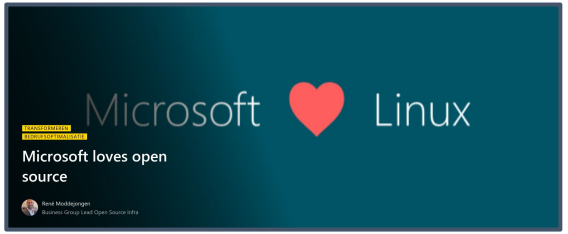
Is it Altruism?

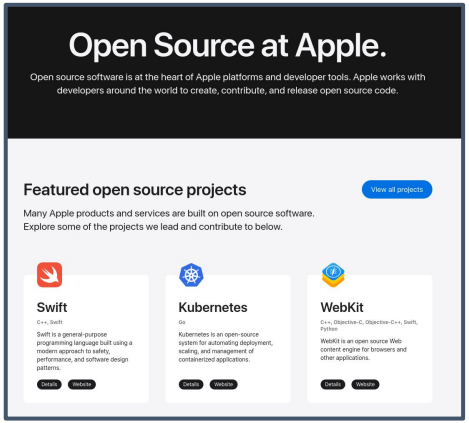












Engineering Economics of Open Source

Open Source transforms engineering economics by offering **cost-efficiency**, **shared innovation**, and **sustainable development** practices.

- Cost Efficiency
  - Lower Upfront Costs: No licensing fees compared to proprietary software.
  - Scalable Resources: Pay for what you need (e.g., cloud Open Source stacks).
- Shared Development Costs
  - Collaboration with the community reduces the cost of building and maintaining
  - Companies share the burden of innovation (e.g., Kubernetes developed by Google, Red Hat, and others).
- Flexibility and Avoiding Lock-In
  - Open Source ensures vendor independence.
  - Adaptable to specific needs without waiting for vendor updates.
- Speed to Market
  - Leveraging existing Open Source projects accelerates development.
  - Example: Adopting TensorFlow or PyTorch for AI solutions can save months of R&D.
- ROI of Contributions
  - Contributing back Reduces long-term costs by:
  - Ensuring compatibility with main codebases.
  - Gaining access to new features and updates.

Side Notes

- It isn’t about altruism – it’s about economics
- It is economically an easier decision to use (borrow)-and-share than build or buy
- You always get more than you give economically

Personal Economics

- Having one's name on key contribution streams in an open source community world is some of the best resumé content one can have:
  - to get work done,
  - to work in a collaborative engineering setting,
  - to demonstrate your understanding of a technology base.
- The connections you make in a community expands your network – and it's the weak connections that make a difference in job market
- Reading software is a good way to improve one's skills, so reading known good software projects is an opportunity to improve skills.

Granovetter, Mark "The Strength of Weak Ties", Stable URL:
https://www.jstor.org/stable/2776392

Let’s Summarize Consumption Economics

- **Build** vs **Buy** vs **Borrow + Share** as the fundamental use economics
- You ALWAYS get more than you give
- When consuming you can easily be gaining 1-3 orders of magnitude of software value
- The economics works from the smallest scale (personal) all the way up to the largest scale (complex projects like compilers and operating systems)

From Consumers to Contributors: How Companies are Driving Open Source Innovation

- Airbnb: Published Apache Superset, now a leading BI tool for data visualization.
- Lyft: Open-sourced Envoy, a high-performance service proxy, powering service meshes like Istio.
- Google: Published Kubernetes, now the de facto standard for container orchestration.
- Netflix: Contributed Spinnaker, a multi-cloud continuous delivery platform.
- Meta (Facebook): Released PyTorch, one of the most popular AI/ML frameworks.
- Twitter: Published Apache Heron, a real-time stream processing engine.
- Databricks: Contributed to Apache Spark, enabling scalable data analytics.
- Shopify: Contributed improvements to Ruby on Rails, ensuring scalability for e-commerce applications.

Open Source Production Economics
(Why Publish? Why Build A Community)

- Kubernetes = 4.6MLoC = ~$630M
- VS Code = 1.2 MLoC = ~$90M
- Each of the communities have an architecture of participation where 50% of the contributions come from OUTSIDE the primary owner (respectively Google and Microsoft)

Rule of Thumb: 1MLoC = 100 staff * 5 years = $75M ($100K base * 1.5 gross up)

Open Source as a Business

Open Source is NOT a Business Model

- Professional services
  - RedHat, SUSE, ..
  - Consulting, training
  - Problems: Ease of use dilemma, renewal rates, service-only competition
- Open core
  - GitLab, Databricks (Apache Spark), MongoDB
  - Problem: Competition on the commercial offerings. E.g., tools
- Software as a service
  - WordPress
  - Problem: Free Riding by Cloud Providers (e.g., Redis vs Azure Redis Cache)
- Branded merchandise, Voluntary donations