CS 4530 Software Engineering

Module 17: Open Source Principles

Jonathan Bell, Adeel Bhutta, Mitch Wand
Khoury College of Computer Sciences

Learning Objectives for this Lesson

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

- Understand the terminology "free software" and explain open source culture and principles.
- Express an educated opinion on the philosophical/political debate between open source and proprietary principles.
- Reason about the tradeoffs of different open source licenses and business model

In the beginning, there was Open Source

- Hardware was not yet standardized, computer vendors focused on hardware innovation, building new operating systems for each platform
- Much software development focused in academic labs, and AT&T's Bell Research Labs
- Unix created at Bell Labs using the new, portable language "C", licenses initially released with source code

IBM 704 at NASA Langley in 1957 (Public domain

The Case Against Open Source

- "Open-Source Doomsday": Once all software is free, we'll stop making more software and have a market collapse
- Innovation will be stifled by the risk that software will be copied
- Making source code public means easier to attack
- "Anarchistic" licensing prevents companies from profiting from open source software

[Variation of popular meme, original source unknown]

The Case For Open Source

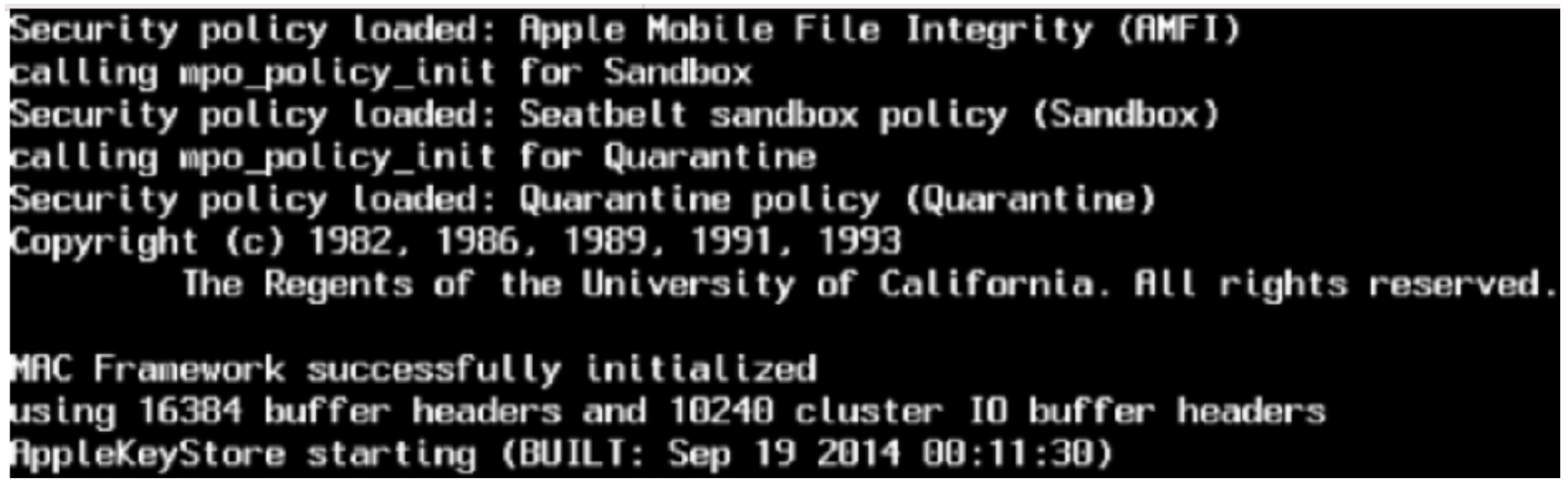
- Many eyes make all bugs shallow
- End-users can improve and customize software to their needs
- New features can be proposed and developed organically
- Greater productivity when more code is reused (easier with open source)

Open. Collaborative. Flexible.

Open Source enables Microsoft products and services to bring choice, technology and community to our customers.

[Screenshot, 2022, opensource.microsoft.com]

UNIX, BSD and GNU



```
Security policy loaded: Apple Mobile File Integrity (AMFI)
calling mpo_policy_init for Sandbox
Security policy loaded: Seatbelt sandbox policy (Sandbox)
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

- 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
- 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”

- Although UNIX was distributed to licensees with source code, the license was still restrictive
- Richard Stallman’s Free Software Foundation - free as in *liberties*
  - 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 redistributed copies (of the original) so you can help others
  - Freedom 3: The freedom to distribute copies of your *modified* version to others

Richard M Stallman (Licensed under GFDL)

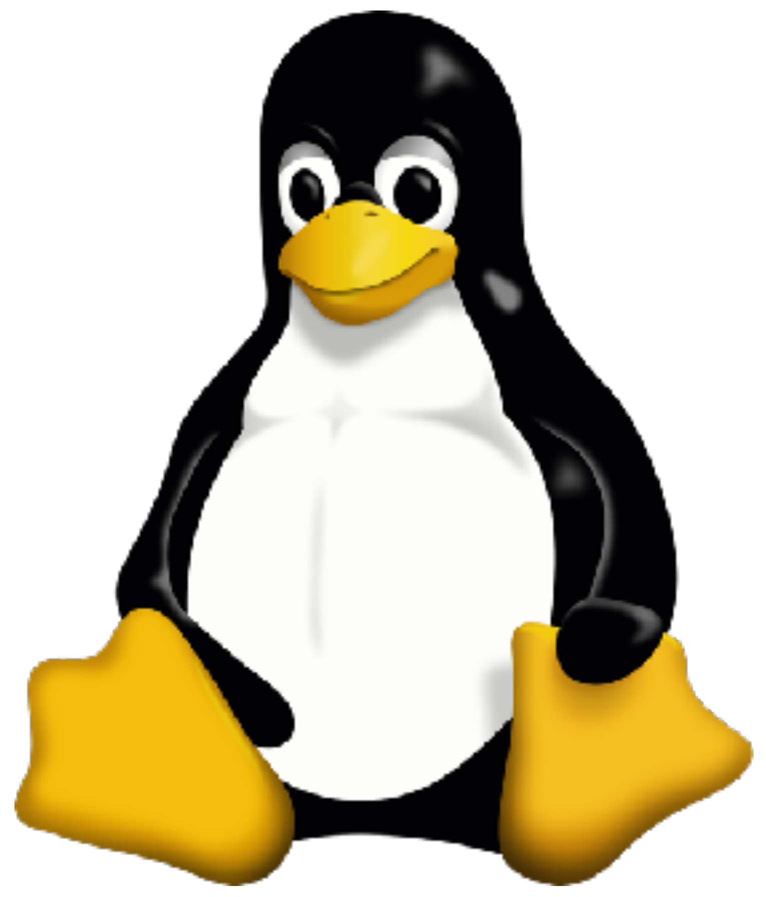
Free Software as a Philosophy

“Free as in Speech, not as in beer”

- Free Software Foundation: Free software should be licensed under the GNU Public License (GPL), considering questions like:
  - Are you *required* to redistribute any modifications (under same license) - “copyleft”
  - Can you redistribute executable binaries, or only source?
  - Are you allowed to use the software in a restrictive hardware environment? (“Tivioization”)
- Popular alternative: “Do whatever you want with this software, but don’t blame me if it doesn’t work” (“freeware”)

GNU/Linux (1991-Today)

- Stallman set out to build an operating system in 1983, ended up building a tremendous set of utilities that are needed by an OS (compiler, utilities, etc)
- Linux is an operating system 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 began adopting and supporting Linux for enterprise customers - IBM committed over $1B; Red Hat and others

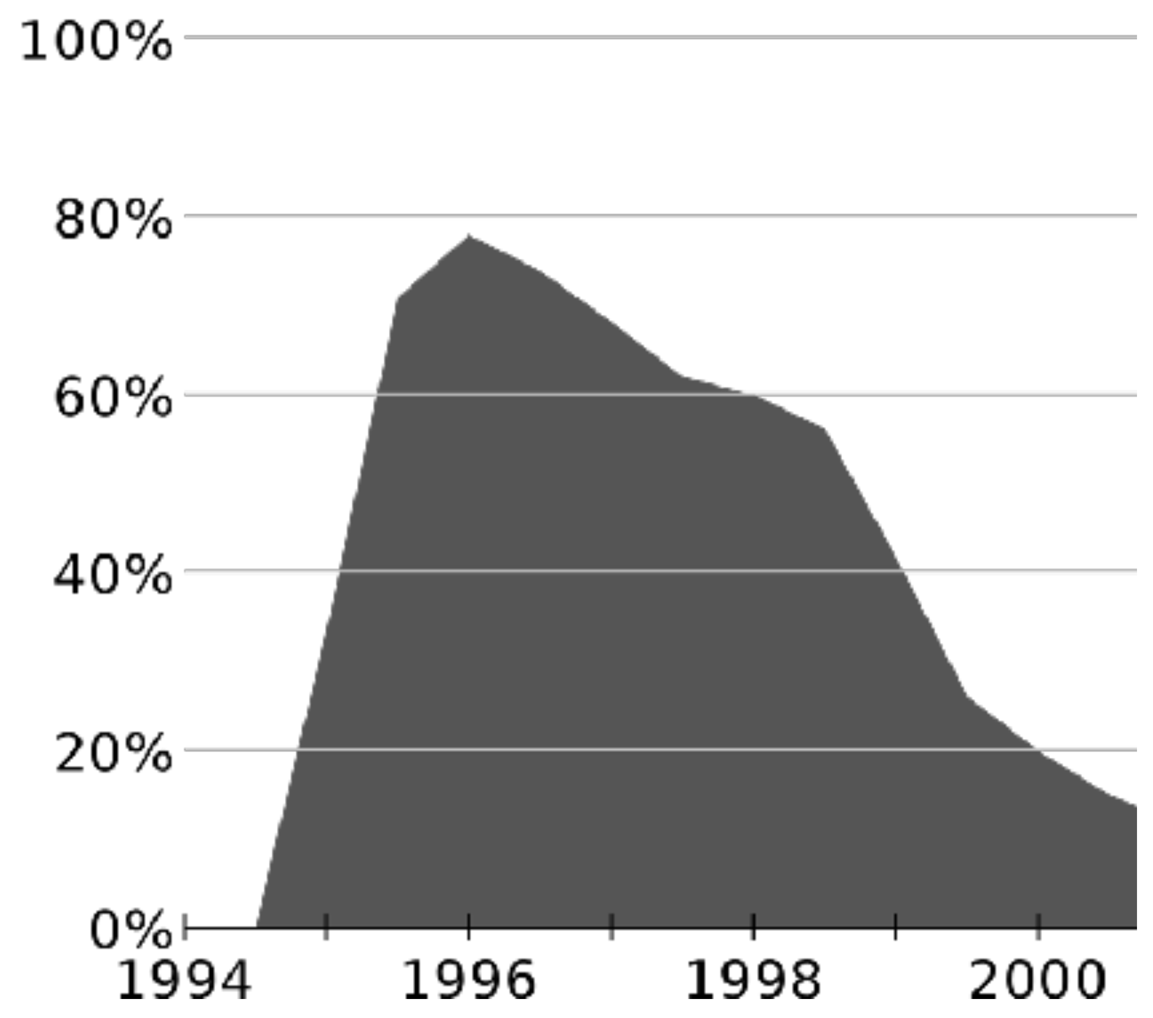


The Cathedral and the Bazaar (1997)

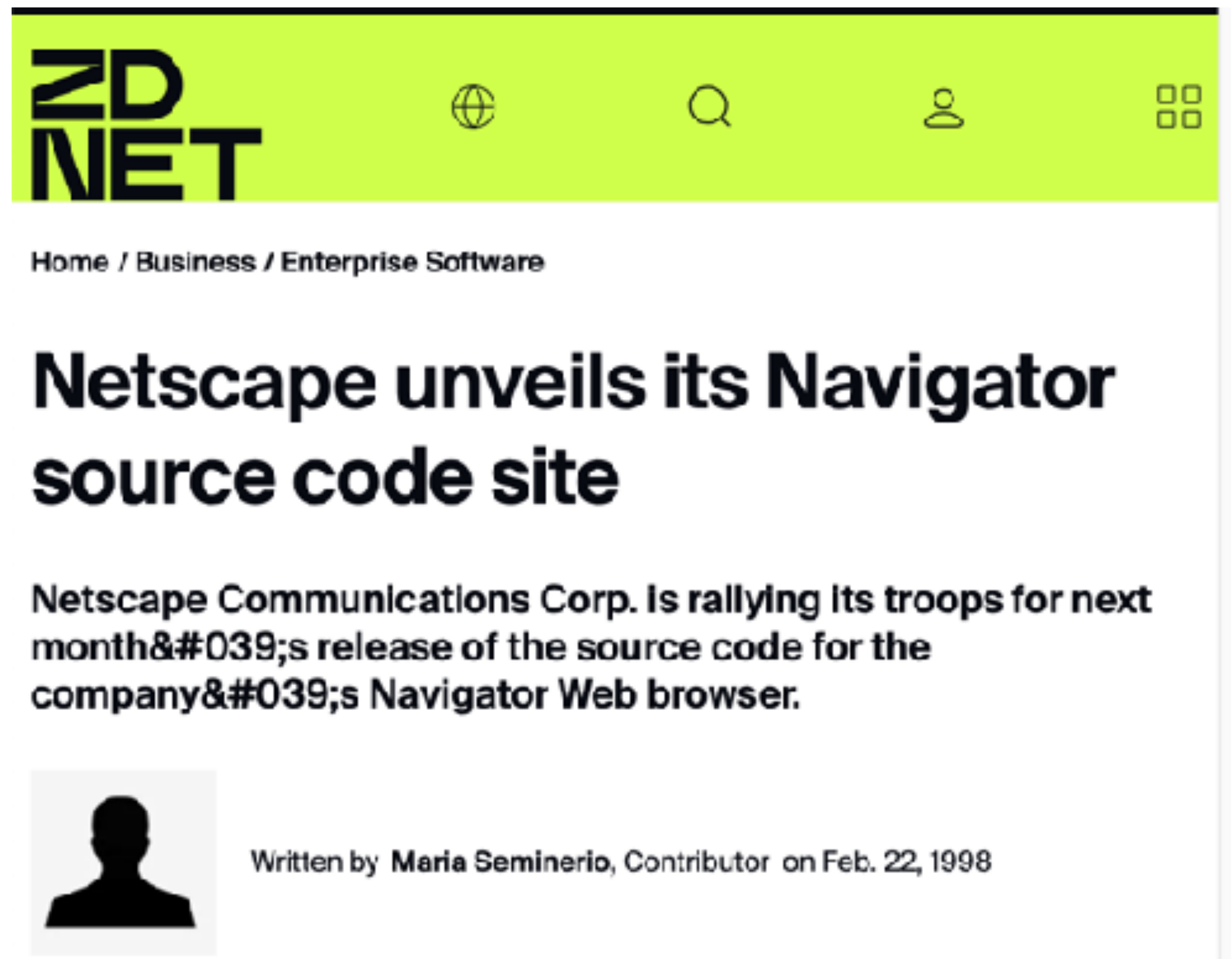
- Eric S Raymond’s 1997 essay compares software development methodologies as a “cathedral” or “bazaar”
- Much OSS today follows this “bazaar” model:
  - Users treated as co-developers
  - Release software early for feedback
  - Modularize + reuse components
  - Democratic organization

Netscape: “Collaborating with the Net”

- Netscape was the dominant web browser in the early 90’s
- Business model: free for home and education use, companies paid to use it
- Microsoft entered browser market with Internet Explorer, bundled with Windows in 1995, soon overtakes Netscape in usage (it’s free, with Windows!)
- January 1998: Netscape becomes first (?) company to make source code for proprietary product open (Mozilla)



Usage Share of Netscape Navigator


ZD NET

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

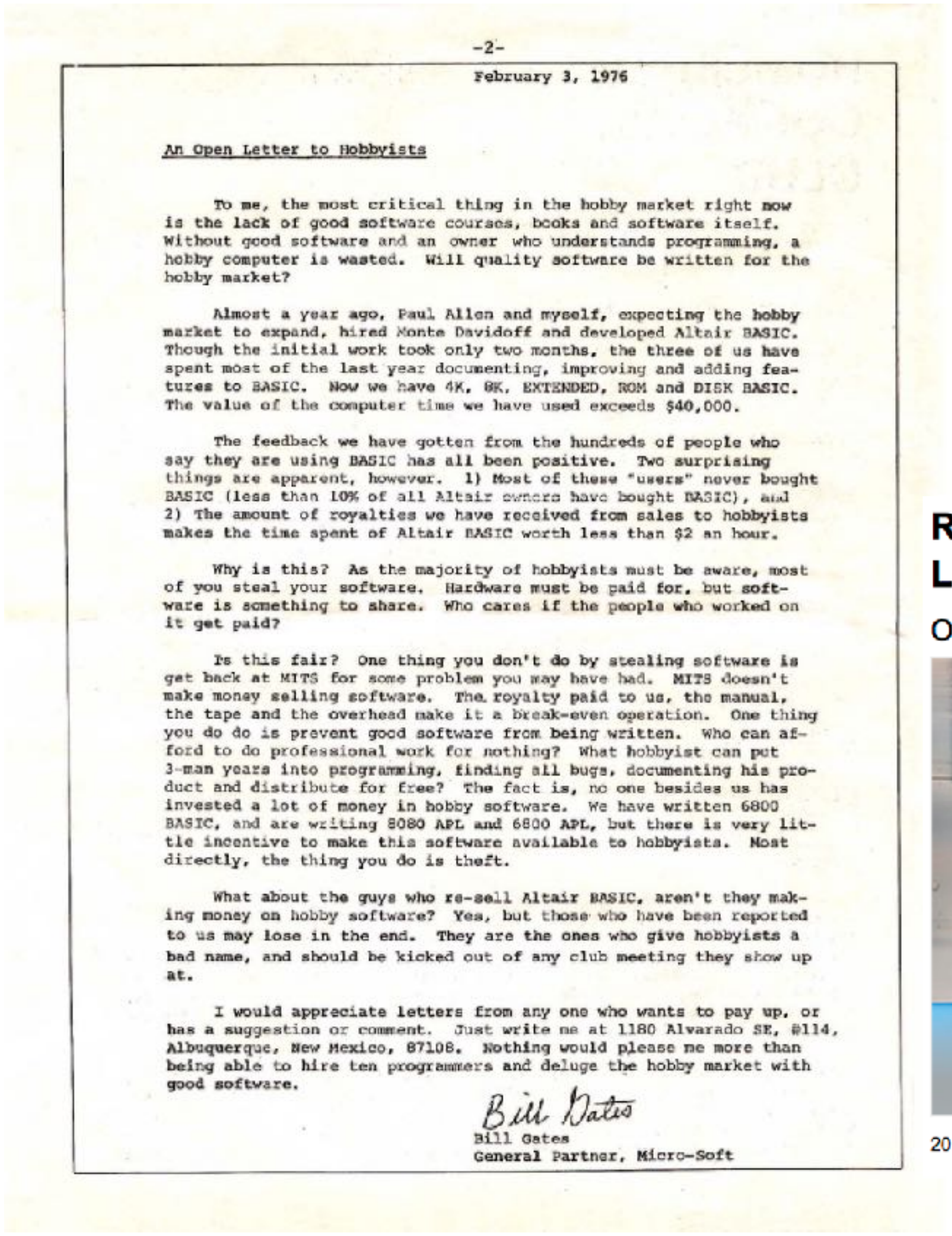
“Open Source”

Open source initiative logo

- Until Netscape/Mozilla, much of open source movement was concentrated in the free software foundation and its GPL
- “Open Source” coined in 1998 by the Open Source Initiative as a term to capture Netscape’s aim for an open development process, Eric Raymond’s “Bazaar”
- 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

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

Is Open Source a Good Business Model?



-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 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

20 Oct 2014 at 23:45, Neil McAllister

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

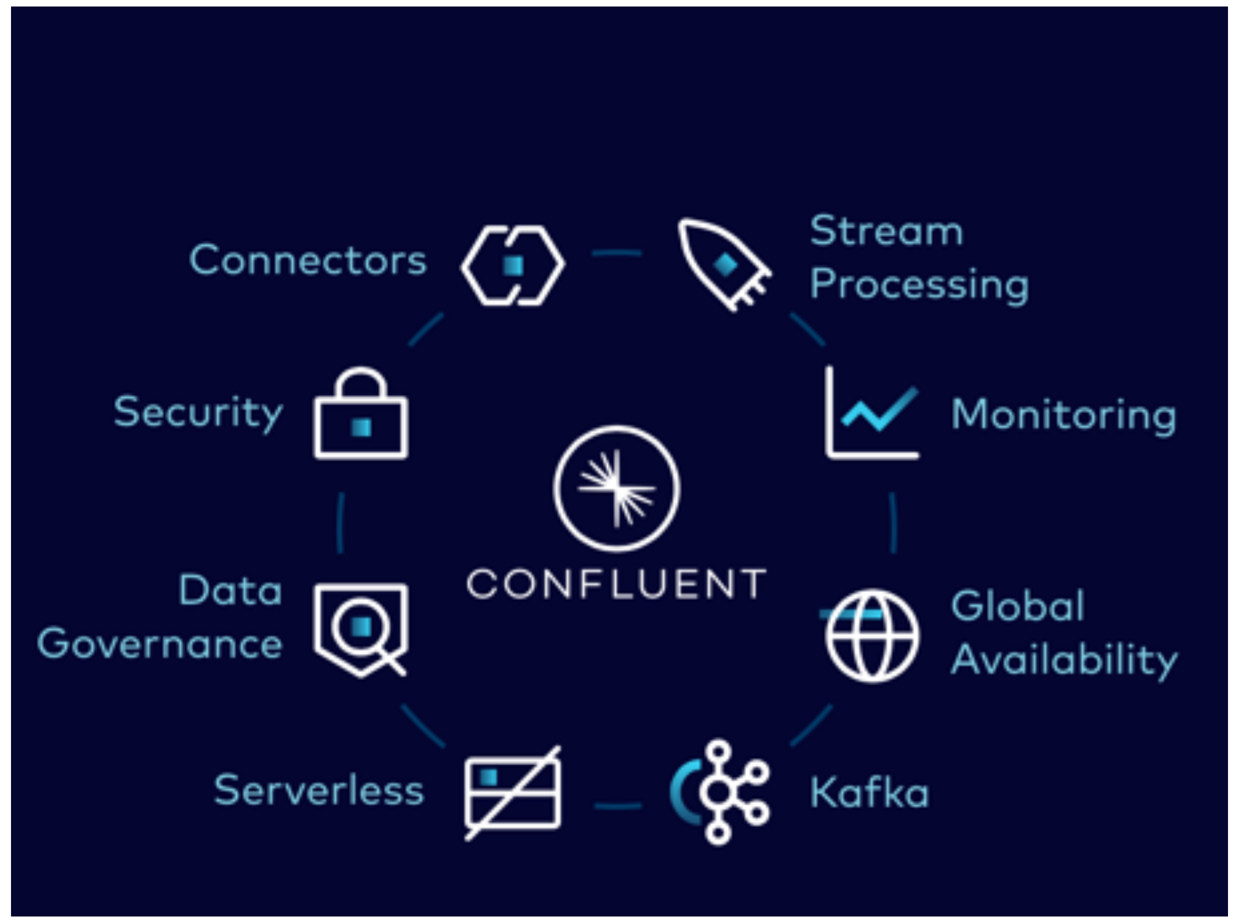
By Steve Lohr
June 4, 2018

IBM TV Commercial: “Prodigy”

https://www.youtube.com/watch?v=x7ozaFbqg00

Model: “Open Core,” closed plugins

- Model: core component of a product is an open source utility; premium plugins available for a fee
- Example: Apache Kafka, a distributed message broker (glue in an event-based system)
    - Product is open source, maintained by Apache foundation, supported by company “Confluent”
    - Confluent provides plugins to connect Kafka to many different systems out-of-the-box



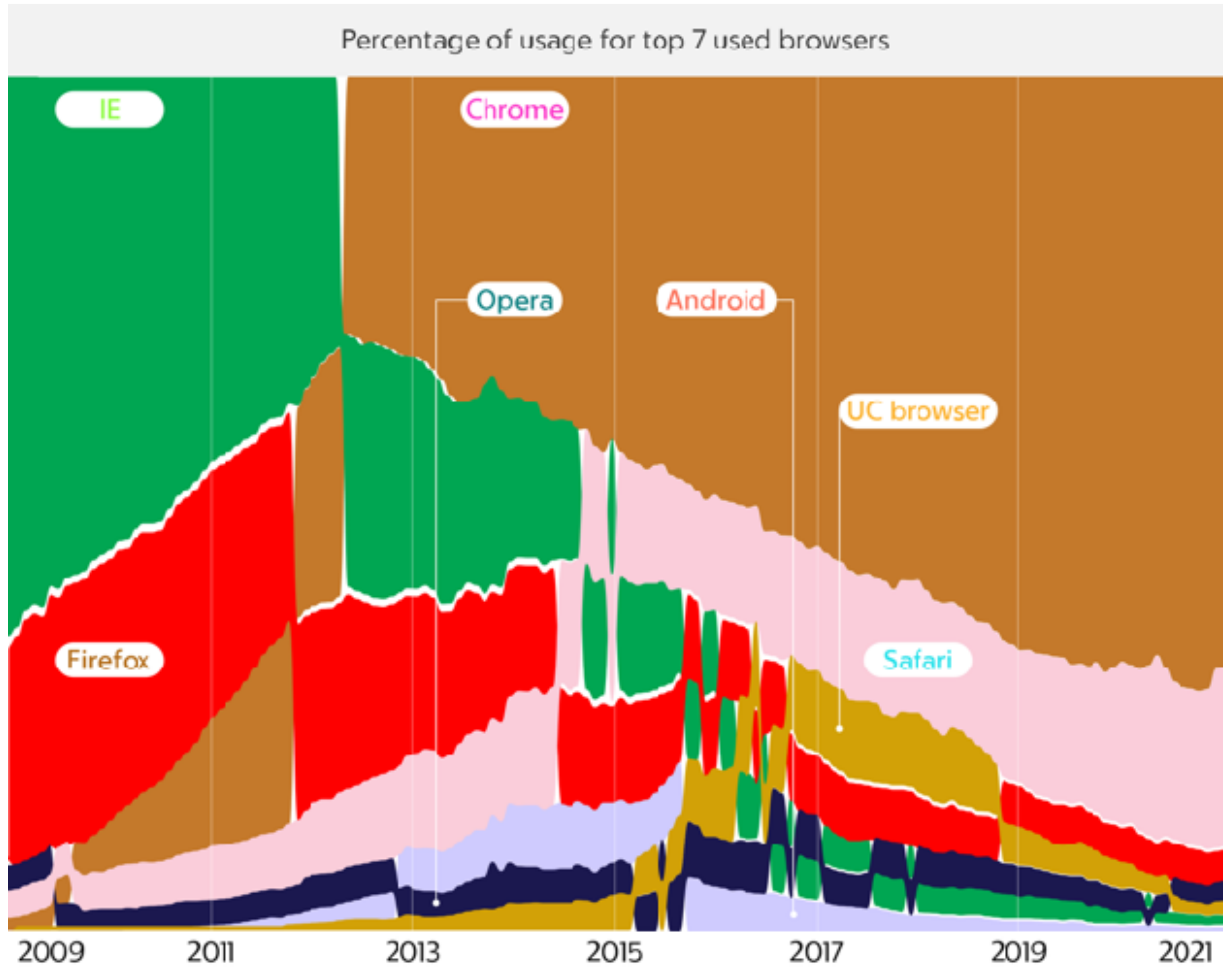
[Screenshot: “Apache Kafka vs Confluent”]

Model: Open Source as a Utility

- 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
- Many companies provide specialized “distributions” of these open source infrastructure and specialized tools to improve them; support the upstream project

The Open Source Browser Wars

- Firefox (based on Mozilla, based on Netscape) is no longer dominant: Chrome and Safari are
- Chrome's core: Chromium (open source)
- Safari's core: Webkit (open source)
- Microsoft's IE successor, Edge? Based on Chromium
- How do browsers differentiate themselves, and why is there still more than one?



[By Datavizzer, CC BY-SA 4.0]

Where do laws come to play in open source?

- Copyright provides creators with protection for 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
- Open source software is generally copyrighted, with copyright retained by contributors or assigned to a foundation/corporation that maintains the product
- Copyright holder can grant a license for use, placing restrictions on how it can be used (perhaps for a fee)
- Common open source licenses: MIT, BSD, Apache, GPL

Licensing: Copyleft vs permissive

- Can I combine some open source software with my product, releasing my product under a different license (perhaps not even open source)?
  - Permissive licenses encourage adoption by *permitting* this practice
  - Copyleft “protects the commons” by *forcing* all linked code to be released under same license (e.g. GPL)
- Philosophy: do we force participation, or try to grow/incentivize it in other ways?

Model: Dual Licensing

- Offer a free copyleft (e.g. GPL) license to encourage broad adoption, prevent competitors from improving it without sharing those improvements.
- Offer custom, more permissive licenses to third parties who are willing to pay for that (e.g. enterprise)
- 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
- Examples: MySQL, Qt

Model: Hosted Open Source Products As A Service

- Model: Creators of open source software provide a cloud hosted, “fully managed” installation of the software, as a service
- Risk: What is your competitive advantage over cloud utility providers e.g. Amazon?
  - Amazon could even make improvements to your GPL code and *not* have to share them because it is not distributing the program (it operates it as a service)
- Example: MongoDB Atlas (document-oriented database)
  - MongoDB created a new license to require copyleft for service providers operating MongoDB as a service
  - Amazon created their own fork of the GPL’ed version of MongoDB

mongoDB

Amazon DocumentDB

Successful Open Source Projects Have Strong Communities

- Open source projects thrive when the community surrounding them contributes to push the project forwards
- 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)

When communities move on: Forks

- When software is released under an open source license, the only rights that the creator can realistically retain are trademarks on name/images - code can otherwise be “forked”
- Example:
  - Sun bought StarOffice in 1999, GPL open-sourced as OpenOffice in 2000 with aim of fighting MS Office
  - 2010: Oracle buys Sun, fires many internal developers, frustrating external community
  - 2011: Community forms a foundation, creates fork LibreOffice, OpenOffice dies off (Oracle transfers to Apache)

Java: Open Source to Retain Control

- While the Java specification is public, there was no open source Java runtime
- Much open source software was/is written in Java, creating “The Java Trap” for open source
- 1996-2006: GNU, Apache (backed by IBM and Apple), and others attempted to create open source implementations; Sun refused to permit these runtimes to be tested for compatibility, prohibiting them from using the term “Java”
- 2007: Sun releases OpenJDK under GPL; third party projects abandoned mostly uncompleted

Android: Build the Ecosystem, not the Operating System

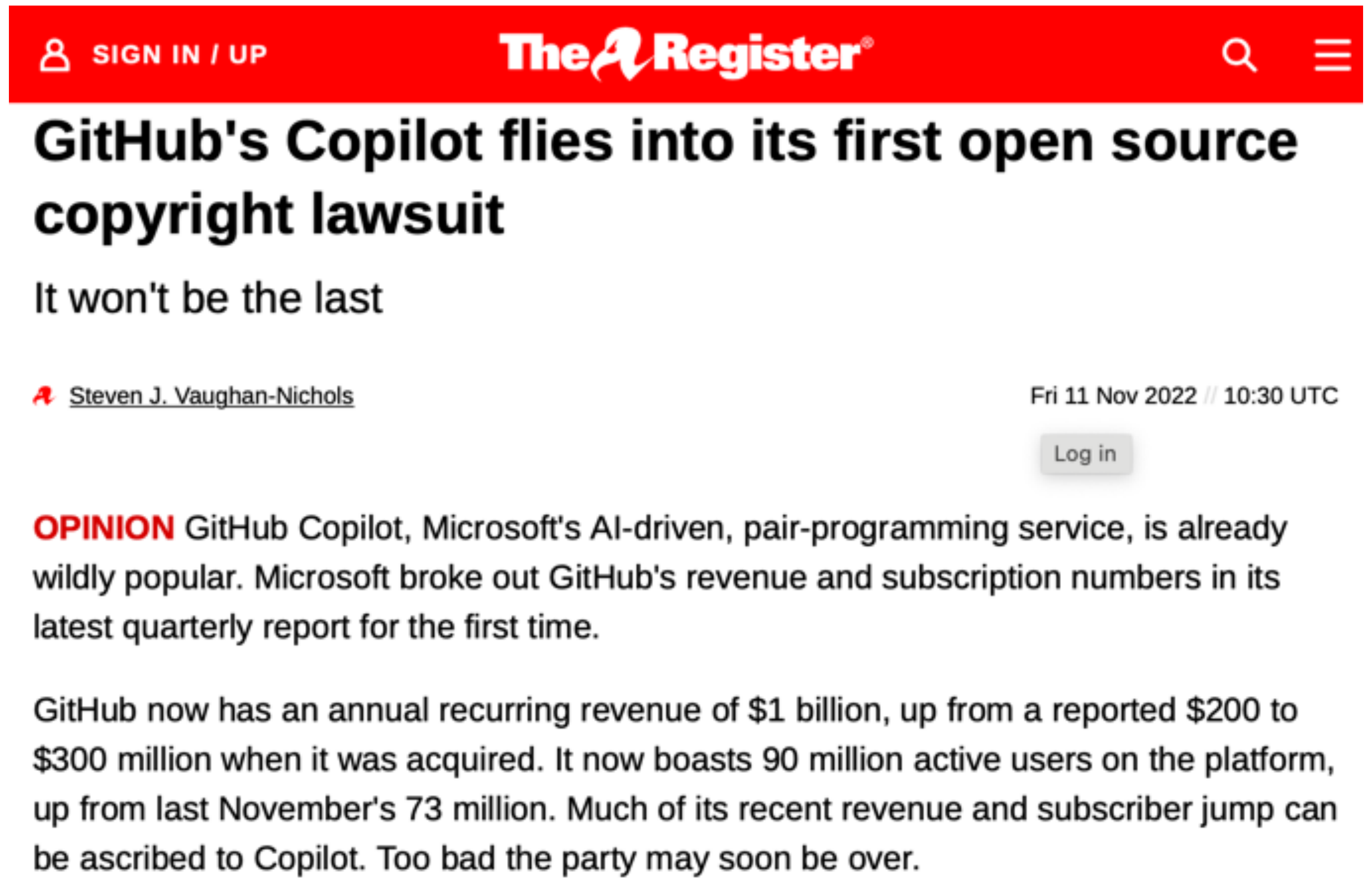
- Model: “Product” is the ecosystem (app store, ads, etc) and the hardware (made by competing manufacturers), *not* the operating system
- Android is entirely open source, built on Linux; applications are written in Java, executed using a custom-built runtime
- To provide implementations of core Java APIs (e.g. java.util.X), Android used the open source Apache Harmony implementations
- Oracle v Google: Oracle asserted that Java APIs were their property (copyright) and Google misused that; judge ruled that could not copyright APIs

Risks Adopting Open Source

- Are licenses compatible? A significant concern for licenses with copyleft:
  - Adopting libraries with copyleft clause generally means what you distribute linked against that library must also have same copyleft clause (and be open source)
  - Including permissive-licensed software in copyleft-licensed software is generally compatible
- Are you certain that the software truly is released under the license that is stated? Did all contributors agree to that license?

GitHub Copilot + Codex

- Codex is a large language model trained on all code in public repositories on GitHub
- Copilot suggests lines of code as you program, based on the Codex model
- Copilot will output entire snippets of code from public GitHub repositories
- What is the ownership and license compatibility of the resulting code?



GitHub's Copilot flies into its first open source copyright lawsuit

It won't be the last

Steven J. Vaughan-Nichols　Fri 11 Nov 2022 // 10:30 UTC

OPINION GitHub Copilot, Microsoft's AI-driven, pair-programming service, is already wildly popular. Microsoft broke out GitHub's revenue and subscription numbers in its latest quarterly report for the first time.

GitHub now has an annual recurring revenue of $1 billion, up from a reported $200 to $300 million when it was acquired. It now boasts 90 million active users on the platform, up from last November's 73 million. Much of its recent revenue and subscriber jump can be ascribed to Copilot. Too bad the party may soon be over.

https://www.theregister.com/2022/11/11/githubs_copilot_opinion/

Review: Learning Objectives for this Lesson

You should now be able to…

- Understand the terminology “free software” and explain open source culture and principles.
- Express an educated opinion on the philosophical/political debate between open source and proprietary principles.
- Reason about the tradeoffs of different open source licenses and business model