Foundations of Software Engineering

Lecture 24: Open Source

Michael Hilton

institute for SOFTWARE RESEARCH

- https://www.youtube.com/watch?v=9sJUDx7iEJw&start_radio=1&list=RD9sJUDx7iEJw

Learning goals

- 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 the open source model on issues like quality and risk, both in general and in a proprietary context.

Administrivia

- Homework 6 task selection and planning due Tuesday Nov 20th

Retrospectives

- What worked well for us?
- What did not work well for us?
- What actions can we take to improve our process going forward?

“Free as in free speech.”

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

Stallman vs. Gates

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

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

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

The Cathedral and the Bazaar

The Cathedral and the Bazaar

- Cathedral (closed source)
  - Top-down design with focus on planning
- Bazaar (open source)
  - Organic bottom-up movement
  - Code always public over internet
  - Linux/Fetchmail stories

Eric Raymond. Essay 1997

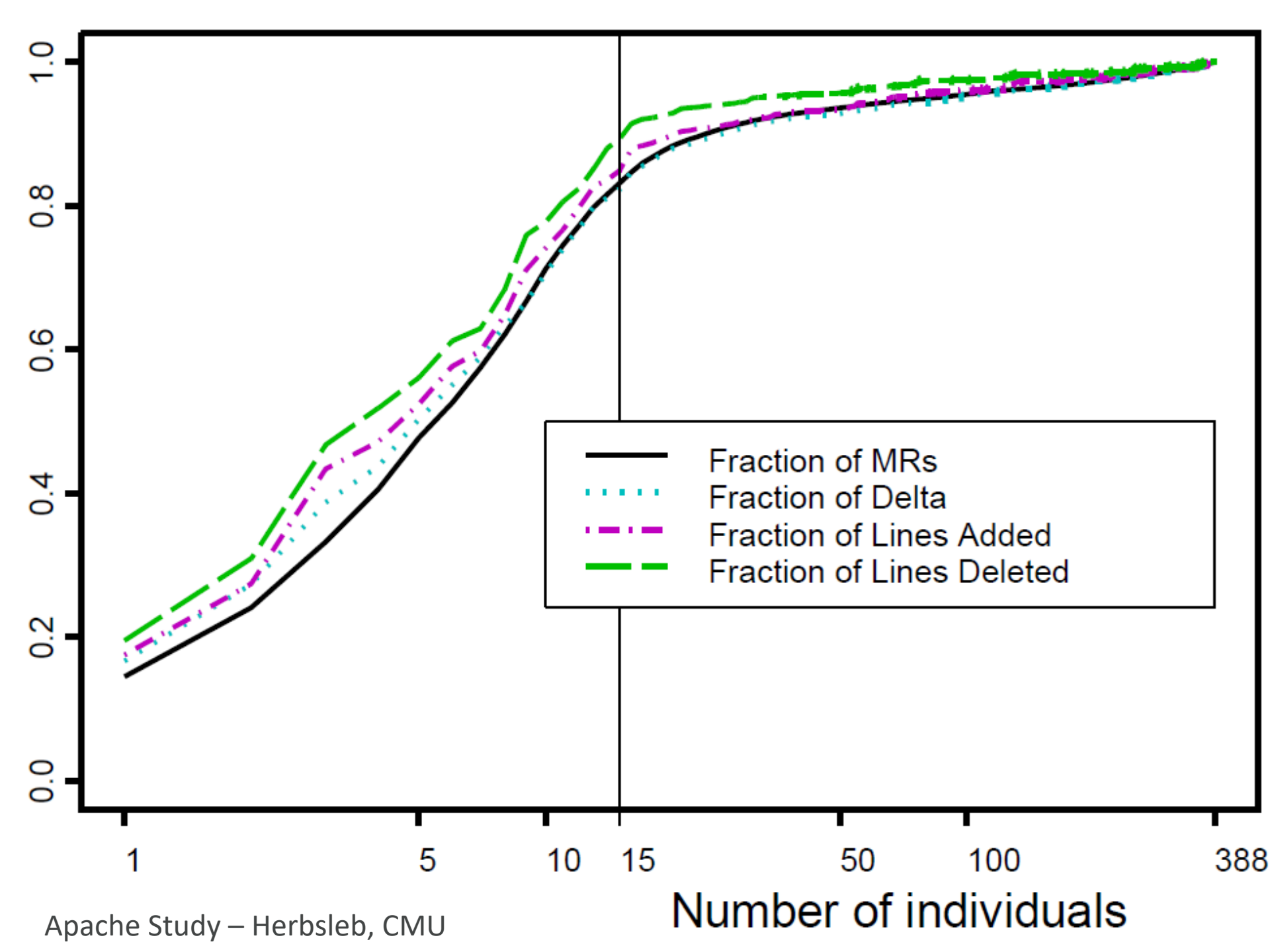
The Cathedral and the Bazaar – Lessons (selection)

- Every good work of software starts by scratching a developer's personal itch
- To solve an interesting problem, start by finding a problem that is interesting to you
- Release early, release often
- Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone
- The next best thing to having good ideas is recognizing good ideas from your users. Sometimes the latter is better.

Eric Raymond. Essay 1997

Open Source Teams

- Potentially open for everybody
- Process to vet contributions
- Typically many contributors but small core teams



Apache Study – Herbsleb, CMU

Social Coding

- Github, Bitbucket, etc.
- Add social networking features to coding
  - Follow users
  - Watch repositories
- Allows team structure to emerge as opposed to previous planning

How do open source programs make money?

- Red Hat – revenues of about $2 Billion last year and is worth approximately $15 Billion.

- Mozilla – has revenues of $300 Million annually

- Apache Software Foundation – recent revenue of $1 Million

Open Source Business Models

- Open source as hobby; resume building
- Selling support/expertise instead of software
  - RedHat
- Selling complementary services
  - Wordpress
- Developers hired as consultants, for extensions

Other Open Source Business Models

- Companies dedicate resources to projects which help them and the community
  - Apache receives donations
- Selling merchandise – Canonical (Ubuntu)
- Selling advertising or customer traffic – Mozilla

Quality?!

"There are no technical requirements for the plugins aside from them being able to be installed on a fresh Eclipse platform. We leave it to the community to find and report bugs related to technical features and conflicts."

-Eclipse Marketplace, Dec 2014

Open Source Famous Phrases

Linus's Law - Many eyes make all bugs shallow

Collaboration over Competition

…is open source code of higher quality?

- How would we be able to tell?

A Case Study of Open Source Software Development: The Apache Server

Measure	Apache	Proprietary System A	Proprietary System C	Proprietary System D
Post-release defects/KLOCA	2.64	0.11	0.1	0.7
Post-release defects/KDelta	40.8	4.3	14	2.8
Post-feature test Defects/KLOCA	2.64	*	5.7	6.0
Post-feature test Defects/KLOCA	40.8	*	164	196

Coverity Report of Open Source

Only tested programs which use Coverity
Defect density: defects per 1,000 lines
Average defect density of 0.69 for open source and 0.68 for proprietary software, surpassing the industry standard of 1 or less

Defect Density Based on Size

	Proprietary	Open Source
500,000-1,000,000 (LOC)	0.98	0.44
1,000,000+ (LOC)	0.66	0.75

[Coverity, 2012, http://www.coverity.com/press-releases/annual-coverity-scan-report-finds-open-source-and-proprietary-software-quality-better-than-industry-average-for-second-consecutive-year/]

Two years later…

- In 2014, open source defect density went down to 0.61 from 0.69 in 2012
- Proprietary defect density went up to 0.76 from 0.68 in 2012

- …verdict?

OPEN SOURCE IN A PROPRIETARY CONTEXT (BENEFITS VS. RISK)

All Our Patent Are Belong To You

Elon Musk, CEO • June 12, 2014

https://www.tesla.com/blog/all-our-patent-are-belong-you

Yesterday, there was a wall of Tesla patents in the lobby of our Palo Alto headquarters. That is no longer the case. They have been removed, in the spirit of the open source movement, for the advancement of electric vehicle technology.

Tesla Motors was created to accelerate the advent of sustainable transport. If we clear a path to the creation of compelling electric vehicles, but then lay intellectual property landmines behind us to inhibit others, we are acting in a manner contrary to that goal. Tesla will not initiate patent lawsuits against anyone who, in good faith, wants to use our technology.

When I started out with my first company, Zip2, I thought patents were a good thing and worked hard to obtain them. And maybe they were good long ago, but too often these days they serve merely to stifle progress, entrench the positions of giant corporations and enrich those in the legal profession, rather than the actual inventors. After Zip2, when I realized that receiving a patent really just meant that you bought a lottery ticket to a lawsuit, I avoided them whenever possible.

At Tesla, however, we felt compelled to create patents out of concern that the big car companies would copy our technology and then use their massive manufacturing, sales and marketing power to overwhelm Tesla. We couldn't have been more wrong. The unfortunate reality is the opposite: electric car programs (or programs for any vehicle that doesn't burn hydrocarbons) at the major manufacturers are small to non-existent, constituting an average of far less than 1% of their total vehicle sales.

At best, the large automakers are producing electric cars with limited range in limited volume. Some produce no zero emission cars at all.

Given that annual new vehicle production is approaching 100 million per year and the global fleet is approximately 2 billion cars, it is impossible for Tesla to build electric cars fast enough to address the carbon crisis. By the same token, it means the market is enormous. Our true competition is not the small trickle of non-Tesla electric cars being produced, but rather the enormous flood of gasoline cars pouring out of the world's factories every day.

We believe that Tesla, other companies making electric cars, and the world would all benefit from a common, rapidly-evolving technology platform.

Technology leadership is not defined by patents, which history has repeatedly shown to be small protection indeed against a determined competitor, but rather by the ability of a company to attract and motivate the world's most talented engineers. We believe that applying the open source philosophy to our patents will strengthen rather than diminish Tesla's position in this regard.

Microsoft Embraces Open Source

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

[FB-Discuss] Project status

Andrey Loskutov loskutov at gmx.de
Wed Nov 2 06:05:20 EDT 2016

https://mailman.cs.umd.edu/pipermail/findbugs-discuss/2016-November/004321.html

- Previous message: [FB-Discuss] Project status
- Next message: [FB-Discuss] Project status
- **Messages sorted by:** [date] [thread] [subject] [author]

Hi all,

TL;DR: I'm really sorry to say, but FindBugs project in its current form is dead.

Longer explanation follows.

Current project setup is:

1) On the plus side, we have two committers with push rights to the github repo, however one from this two (Tagir) is not active anymore for the project and second one (me) has no free time to work on the project. That's however all about the good things...

2) Only the project leader Bill Pugh has admin rights for the project web page and the github project group and page. We cannot deploy any website update, we can't add new project members, we can't manage code access rights, we can't publish releases to the well known update sites without his help. Without him, we have no admin rights to anything, we can only push to the repository.

3) It looks like Bill Pugh is not interested in the FindBugs project anymore, and we can't reach them. I say "it looks like" because we requested his help for the project many times (via direct mails, postings to the list and to the github issues) but haven't received any sign of life from him since a year. We know that he is active elsewhere (https://twitter.com/wpugh). A week ago I've sent another mail to Bill (and CC to the findbugs-core at lists.sourceforge.net mailing list) asking him about the current project state - with no answer so far. You can read my mail in the attachment. IMHO no answer to this mail was the answer enough. Either Bill has completely lost access to his old mail account (which is possible too) or he is ignoring me or the project (which is more likely).

If someone has a possibility to contact him in some way (twitter/mail/phone/whatever) and point him to the discussion on this list - please do so!

Without Bill Pugh FindBugs project is headless and effectively *finally* dead. It is not the *only* reason for the project to be dead, but a bigger one, and the last one.

The other major reasons for the FindBugs current bad state:

1) The code is very complex, has "organically grown" over a decade, is not documented and has poor public interfaces. Most of the code consists of the very low level bytecode related stuff, tightly coupled with the ancient BCEL library, which doesn't scale and is not multi-thread safe. No one enjoys maintaining this code, at least not me. I see no future

Y **Hacker News** new | threads | comments | show | ask | jobs | submit

▲ FindBugs project in its current form is dead (umd.edu)

264 points by fanalin 24 days ago | hide | past | web | 116 comments | favorite

▲ billpugh 23 days ago [-]

FindBugs isn't dead (although my participation had been in hibernation for a while).

I've been juggling far too many projects, but I'm now working to move FindBugs back into the active rotation.

I also want announce I'll be working with GrammaTech as part of the Swamp Project, and they will be helping with rebooting the FindBugs project. year), and although I've known that GrammaTech was likely to win an award, this hasn't been official and something I could talk about until recen concrete to talk about as far as that goes; but I don't yet have the information I wanted to share.

Thanks to all the FindBugs fans and supporters who lobbied for me to return to active maintenance of FindBugs. Give me a week to get up to spee

Bill Pugh

▲ unreal37 23 days ago [-]

It's still not a good sign that it took this level of public attention to get you to reply to the active community on their urgent needs.

▲ dmuth 23 days ago [-]

^ this. There is absolutely no reason to not be answering emails, even if to say, "I'm really swamped, and need help."

I don't mean to denigrate you, but I must be candid here: hoarding admin rights so that only you have them and no one else can get

Going forward, I would recommend taking a look through other projects you may be involved with, and make sure that you are not th increase the "bus factor" to greater than 1: https://en.wikipedia.org/wiki/Bus_factor

Good luck.

▲ junkster78 23 days ago [-]

Totally agree. Even worst, the community has already forked and is getting aligned behind https://github.com/spotbugs/spotbu stays forked, with split efforts and APIs diverging forward, or it's shutdown to trust Bill again and hope for the best.

I for one have lost all trust on his ability to lead the project after such long absence, and the fact that he only reappeared after more about what people may be thinking / saying of him, than the well being of FindBugs.

I fear, the only way for FindBugs to stay alive, is for Bill to do what he probably should have done a long time ago: step down.

▲ issaria 22 days ago [-]

This is also the problem for other projects, the leaders' unwilling to giveup their position even they are inactive for a ver

Open SSL/Heartbleed.

- In 2013, OpenSSL made $2,000 in donations (and some from other sources)
- One full time programmer
- Heartbleed (2014): Vulnerability was found that effected about 17.5% of web servers (half a million)
- Used by Yahoo, Twitter, Google
- Who is responsible?

Case Study: OpenSSL

- When HeartBleed occurred, Google reported the bug and later submitted a patch
- After the HeartBleed bug, more than 17 companies agreed to each contribute $100,000 annually for 3 year to the Core Infrastructure Initiative.
- Core Infrastructure Initiative distributes funds to needy but important projects

Bug Bounties

- Facebook, Google, Yahoo, Microsoft, and other companies have rewards for finding bugs and reporting them
- Usually $100 or more for simple bugs and higher rewards for more serious bugs
- Bounties can save the company from malicious exploits, which can cost the company much more.
  - Ponemon Institute reports average cost of $3.79 million per company data breech (2014)

Risks of *not* open sourcing something?

Proprietary methods to gain community benefits

- Release early, release often; Continuous or small updates instead of big version changes
- “Many eyes make all bugs shallow”
- Recognize good ideas from your users.
- Collaboration over competition
- Promote users to report bugs and monitor new releases (easier if using software as a service)
- Allow users to write mods for the product (usually in a controlled way) or promote feature requests

ONE MORE RISK IN PROPRIETARY CONTEXT: LICENSES

Why learn about licenses?

- Companies will avoid certain licenses – commonly the copyleft licenses
- Specific licenses may provide competitive advantages
- You may eventually want to release open source software or become more involved in an open source project

Open Source Licenses

Software	Percentage
MIT License	24%
GNU General Public License (GPL) 2.0	23%
Apache License 2.0	16%
GNU General Public License (GPL) 3.0	9%
BSD License 2.0 (3-clause, New or Revised) License	6%
GNU Lessor General Public License (LGPL) 2.1	5%
Artistic License (Perl)	4%
GNU Lesser General Public License (LGPL) 3.0	2%
Microsoft Public License	2%
Eclipse Public License	2%

List from: https://www.blackducksoftware.com/resources/data/top-20-open-source-licenses

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)

GPL 2.0 and 3.0 – Addresses free software problems

- 2.0 - Court ruling cannot nullify the license and if a court decision and this license contradict in distribution requirements, then the software cannot be distributed
- 3.0 – patent grant and prevent Tivoization
- Not compatible with each other; Can't copyleft both at the same time – phrase: "GLP Version 3 or any later version"

Why would projects choose one license over another?

Apache

A permissive license that also provides an express grant of patent rights from contributors to users.

Required
- License and copyright notice
- State Changes

Permitted
- Commercial Use
- Distribution
- Modification
- Patent Use
- Private Use
- Sublicensing

Forbidden
- Hold Liable
- Use Trademark

View full Apache License 2.0 license »

GPL

GNU Affero GPL v3.0 | GNU GPL v2.0 | **GNU GPL v3.0**

GPL is the most widely used free software license and has a strong copyleft requirement. When distributing derived works, the source code of the work must be made available under the same license.

Required
- Disclose Source
- License and copyright notice
- State Changes

Permitted
- Commercial Use
- Distribution
- Modification
- Patent Use
- Private Use

Forbidden
- Hold Liable

[From http://choosealicense.com/licenses/]

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

MIT License

- Must retain copyright credit
- Software is provided as is
- Authors are not liable for software
- No other restrictions

LGPL

- Software must be a library
- Similar to GPL but no copyleft requirement

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

Apache License

- Apache
  - Similar to GPL with a few differences
  - 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

Perception:

- Anarchy
- Demagoguery
- Ideology
- Altruism
- Many eyes

Open Source Reality

- Aggressive collaborative tool use
  - version control, CI, issue tracker, reviews, …
- Careful management of people
- Process rigor
- Often aimed at expert users

similar to industrial practices

- Intellectual property
- Often industry supported
- Often addressing common assets