Whither Peer Production?

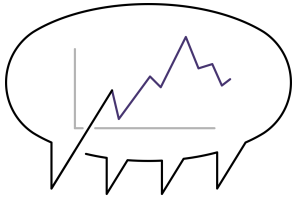
The Shifting Landscape of Online Cooperation

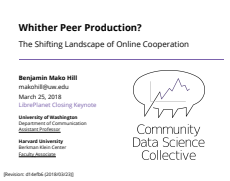
Benjamin Mako Hill
makohill@uw.edu

March 25, 2018
LibrePlanet Closing Keynote

University of Washington
Department of Communication
Assistant Professor

Harvard University
Berkman Klein Center
Faculty Associate





- Thanks the organizers for an amazing conference.
- Thanks the speakers! And attendees!
- Although closing keynotes often try to end on a high note, this talk is about the thing that has me most worried about the future of free software.
- I also want to say that these are new ideas. You are the first people outside my research group to hear them and, although I suspect getting through this is going to my full time here, I really look forward to talking with everybody afterward.

Overview

1. Couchsurfing and Airbnb
2. History of digital commons in three parts:
   - I We built a social movement
   - II We invented new forms of mass collaboration
   - III The proprietary world learned from us
3. The future of free software

Overview

1. Couchsurfing and Airbnb
2. History of digital commons in three parts:
   - I We built a social movement
   - II We invented new forms of mass collaboration
   - III The proprietary world learned from us
3. The future of free software

1. Story from my research: Couchsurfing and Airbnb
2. History of digital commons in three parts:
   - I We built a social movement
   - II We invented new forms of mass collaboration
   - III The proprietary world learned from us
3. The future of free software

CouchSurfing is a website that connects people who need a place to stay with other random strangers they meet on the Internet who have a spare guest room or a couch.

Couchsurfing was created in 2003 as a nonprofit. It has recently claimed to have somewhere 400,000 active hosts (i.e., people who are willing to host others and have published this information on the site).

- How many people stayed in a place they arranged on Couchsurfing last night?
- How many have ever done it?

Airbnb is also a website connects people who need a place to stay with random strangers on the Internet who can help them out.

Airbnb was created in 2008 (5 years after CS). The numbers I found (nearly two years old!) suggested more than 2 million active listings.

- How many people stayed in a place they arranged on Airbnb last night?
- How many have ever done it?

Citation: https://www.quora.com/How-many-hosts-are-there-on-Airbnb

Although CS and Airbnb have very similar high-level goals and work extremely simialrly, they are critically different in one fundamental aspect...

Couchsurfing explicitly bans the exchange of money while Airbnb is built around it.

Couchsurfing explicitly bans the exchange of money while Airbnb is built around it.

"Couchsurfing explicitly bans the exchange of money while Airbnb is built around it."

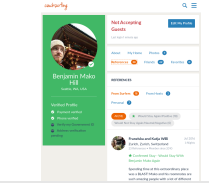
Couchsurfing is a cooperative community. It's a commons for hospitality. On CS people can give and take hospitality freely but they can't buy and sell it.

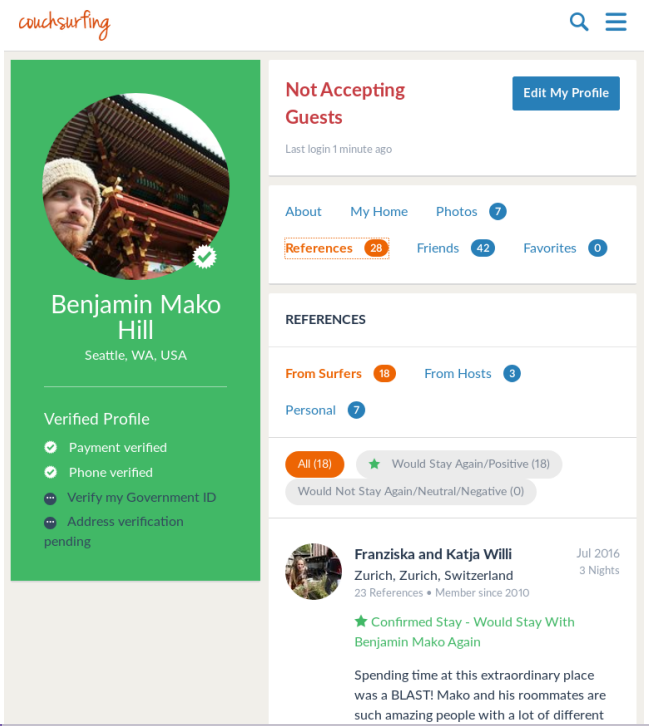
On the other hand, Airbnb is a market. It's people selling hospitality.

I want to reflect briefly on how shocking the success of both platforms is.

Can you imagine if I told you, 20 years ago, that millions of people will be sleeping in the homes of total strangers they had just become acquainted with over the Internet? A a good chunk of this conference would be actually doing it. I think it's clear that it couldn't have happened 20 years ago.

And the reason it is happening now is because of real innovations. Innovations created first Couchsurfing and similar social-based systems like Hospitality Club (which predated it).

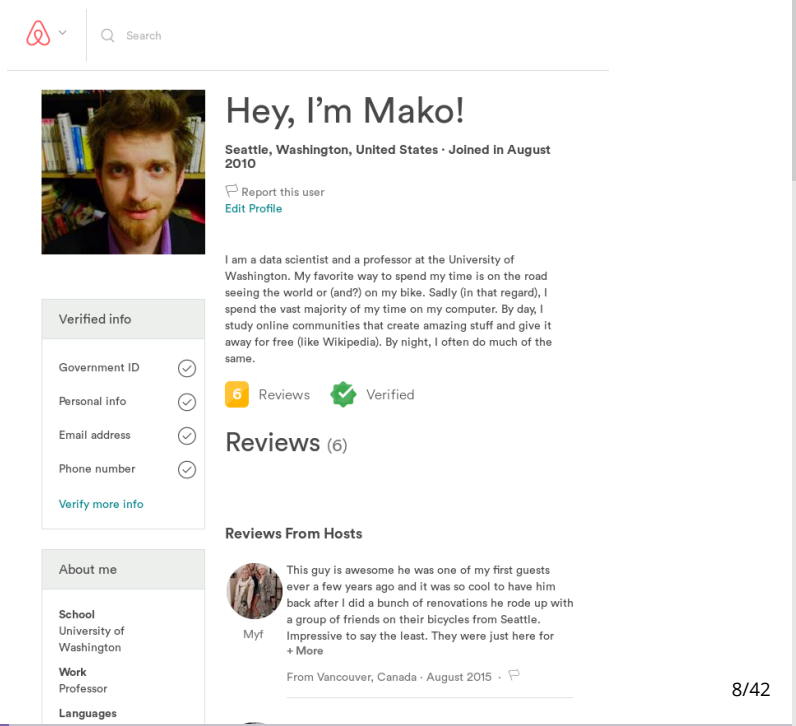


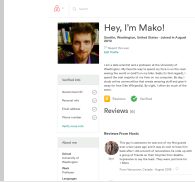


This is my CS profile. We can walk through these innovations:

- Folks upload many personal photos
- Describe their interests in an about page.
- You can register your identity and location (these actually came much more recently).
- And the real reason: There are tons and tons of references from other people who, although also probably strangers, have references themselves. And you can follow them.

It may seem obvious now but this is the technological and social infrastructure that makes staying with random strangers you meet on the Internet possible.

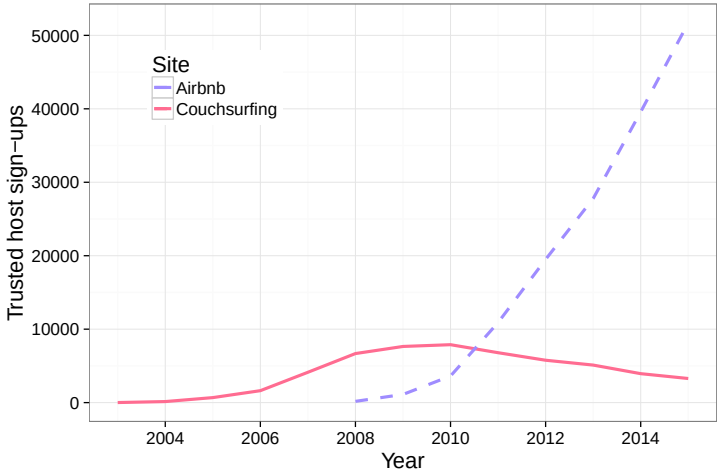




Let's look at my Airbnb profile. It looks extremely similar:

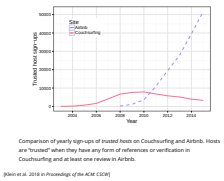
- Photos, descriptions of interests, etc.
- And a network of references.

Airbnb, founded 5 years after CS, took the tools from CS and its predecessors and put them to work in a market and money-based context. Airbnb's success is built on the fact that it has adopted the innovations, created by a non-commercial socially-produced commons. They learned from Couchsurfing.



Comparison of yearly sign-ups of *trusted hosts* on Couchsurfing and Airbnb. Hosts are "trusted" when they have any form of references or verification in Couchsurfing and at least one review in Airbnb.

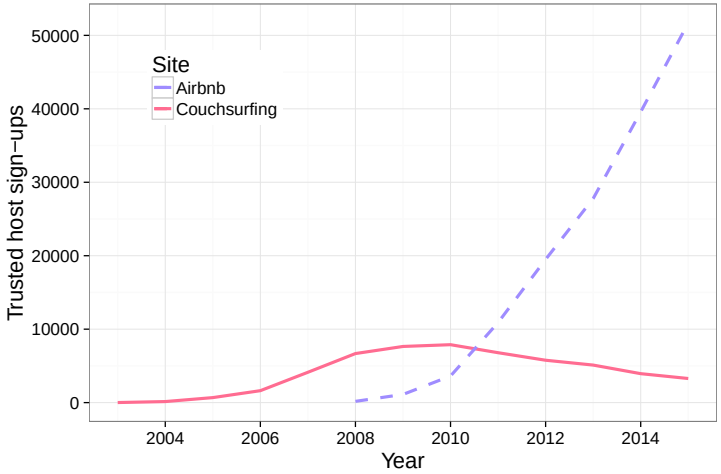
[Klein et al. 2018 in *Proceedings of the ACM: CSCW*]



This is a graph from that paper I published with Max Klein (grad student at University of Minnesota).

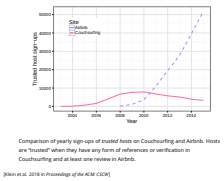
Very roughly, it shows number of validated hosts on each site based on when they signed-up.

Although Couchsurfing is much older than Airbnb, it appears to have been in slow decline in terms of new sign-ups. Strikingly, its peak is contemporaneous with the beginning of Airbnb's rapid growth.



Comparison of yearly sign-ups of *trusted hosts* on Couchsurfing and Airbnb. Hosts are "trusted" when they have any form of references or verification in Couchsurfing and at least one review in Airbnb.

[Klein et al. 2018 in *Proceedings of the ACM: CSCW*]

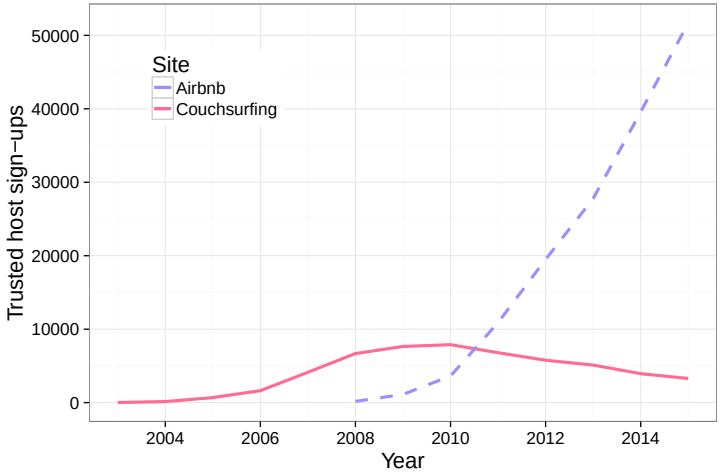


It is tempting to read this graph as evidence of Airbnb killing CouchSurfing. After all, you can only one person can sleep on your couch on a given night and many people will, given the presence of both options, chose the one that brings in cash.

And this has absolutely happened. We talked with people, quoted in our paper, who said exactly this kind of thing.

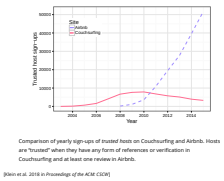
But it seems clear to me, and this is just intuition, that most people hosting on Airbnb would never have hosted people for free on a site like Couchsurfing.

The bigger takeaway, I think, is not about Couchsurfing's decline (or maturation, perhaps) but about Airbnb's incredible growth.



Comparison of yearly sign-ups of *trusted hosts* on Couchsurfing and Airbnb. Hosts are “trusted” when they have any form of references or verification in Couchsurfing and at least one review in Airbnb.

[Klein et al. 2018 in *Proceedings of the ACM: CSCW*]



In 2007, before Airbnb was created, we thought that the only way to get people on the Internet to open their homes to strangers at scale was to do so in a commons.

Airbnb is an example of how a non-commons-based producer has learned from commons-based ones in ways that allow them to get many of the benefits of working in a commons without actually providing goods or services that are free as freely shared, or for that matter free as in beer.

With those benefits, plus all the benefits that markets bring, the non-commons-based producer is increasingly beating the commons.

I believe that the key to understanding what has happened here involves first understanding this history of how we got to where we are today and where we are going.

A history of online cooperative production in three “eras”

Era I: Social movements for knowledge commonses

A history of online cooperative production in three “eras”
Era I: Social movements for knowledge commonses

To build up this understanding, I’m going to tell you all the story of online cooperative production in three parts.

“Era I” will be quickest because I realize that this is perhaps the room of people, on earth at this moment, who I least need to explain that free software began as, and continues to be, a social movement—not just a way of developing software.

"Extracting money from users of a program by restricting their use of it is destructive because the restrictions reduce the amount and the ways that the program can be used. This reduces the amount of wealth that humanity derives from the program."

—GNU Manifesto

"Extracting money from users of a program by restricting their use of it is destructive because the restrictions reduce the amount and the ways that the program can be used. This reduces the amount of wealth that humanity derives from the program."

—GNU Manifesto

As you all known, like any good movement, we can trace our roots back to a manifesto. The GNU manifesto, that lays out the need for our movement.

> *"Extracting money from users of a program by restricting their use of it is destructive because the restrictions reduce the amount and the ways that the program can be used. This reduces the amount of wealth that humanity derives from the program."*

Software doesn't mind if it's not-free. But users mind quite a lot! The manifesto reminds us that restricted software is bad because it restricts what users can and cannot do.

Insofar as our software frames the way we understand the world and the ways we interact, the question of who controls it is an enormously important political question. And our answer to that question is that every user must control their own software.

"Extracting money from users of a program by restricting their use of it is destructive because the restrictions reduce the amount and the ways that the program can be used. This reduces the amount of wealth that humanity derives from the program."

—GNU Manifesto

"Extracting money from users of a program by restricting their use of it is destructive because the restrictions reduce the amount and the ways that the program can be used. This reduces the amount of wealth that humanity derives from the program."

—GNU Manifesto

The most surprising thing to me, rereading the GNU manifesto recently, is not it's eloquent articulation of the importance of software freedom.

The most surprising thing to me was that it seemed to assume that GNU would be built more or less in the centralized way that proprietary software had traditionally been built.

"I have found very many programmers eager to contribute part-time work for GNU. For most projects, such part-time decentralized work would be very hard to coordinate; the independently written parts would not work together."

—*GNU Manifesto*

"I have found very many programmers eager to contribute part-time work for GNU. For most projects, such part-time decentralized work would be very hard to coordinate; the independently written parts would not work together."

—*GNU Manifesto*

RMS wrote:

"I have found very many programmers eager to contribute part-time work for GNU. For most projects, such part-time decentralized work would be very hard to coordinate; the independently written parts would not work together."

The GNU manifesto voices real skepticism about part-time work and decentralized collaboration.

It's not that just that we didn't realize that developing software in freedom would be easier or better. We often assumed that we would be at a disadvantage to proprietary production!

"The real reason programmers will not starve is that it will still be possible for them to get paid for programming; just not paid as much as now."

—GNU Manifesto

"The real reason programmers will not starve is that it will still be possible for them to get paid for programming; just not paid as much as now."

—GNU Manifesto

The manifesto says:

> *"The real reason programmers will not starve is that it will still be possible for them to get paid for programming; just not paid as much as now."*

A social movement was necessary because we felt that software commons would be good for society but, in general, bad for the would-be programmer participants would stood to benefit personally from treating the software as their intellectual property.

We built our software and created new ways of producing it even when we thought it was going to be worse and we thought that we were going to get paid less.

A history of online cooperative production in three “eras”

Era II: Peer production

By the mid-nineties, we realized that we had been wrong about getting paid less and about being less effective.

We had discovered that building software in freedom was not as just as good as doing in the bad old way. It was, in some cases, better.

Most strikingly, the free software movement (and the free culture movement inspired by it) had created the two largest and important products of mass-collaboration in history.

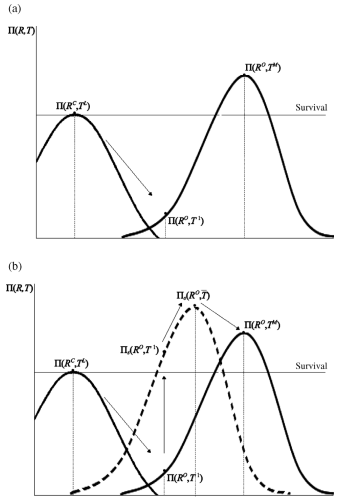
By 2004 or so, GNU/Linux had destroyed a billion dollar decades old industry producing operating software for servers.

At that point, Wikipedia was already the largest, most comprehensive, and highest quality encyclopedia in history and the fifth most visited website in the world.

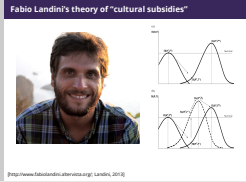
And there was no firm or market behind either one.

Economists couldn’t for the life of them figure out what had just happened.

Fabio Landini's theory of "cultural subsidies"



[http://www.fabiolandini.altervista.org/; Landini, 2013]



Fabio Landini is one economist who provided an answer

Landini created an theoretical model, tons of math, that shows how working openly and providing users with freedom can be at least as good a way of building software as working without it.

But the most critical problem for Landini's theory was explaining the emergence of this new form of collaboration based in freedom: If you already knew how to develop proprietary software in a way that was profitable, why on earth would anybody spend the time to discover a new and potentially better way?

Landini could only explain the emergence of a better development model, driven by freedom, if a group of people were motivated to put time and energy into innovating and experimenting when the model wasn't better. He called this "extra" effort a cultural subsidy.

Landini argued the free software movement “subsidized” the discovery of peer production.

Landini argued the free software movement “subsidized” the discovery of peer production.

The subsidy that allowed the world discover peer production was provided by ideological motivated free software activists.

We built our software and created new ways of producing it even when we thought it was going to be worse and we thought that we were going to get paid less.

Peer Production

New modes of collective production made possible by lowered transaction costs through new communication technologies. (Benkler 2003, 2006)

1. Goals are set and executed in a decentralized manner.
2. Participants have a diverse range of typically non-monetary motivations.
3. Projects are governed as open commons.

[Benkler, et al., 2016; Benkler, 2015]

Peer Production
New modes of collective production made possible by lowered transaction costs through new communication technologies. (Benkler 2003, 2006)
1. Goals are set and executed in a decentralized manner.
2. Participants have a diverse range of typically non-monetary motivations.
3. Projects are governed as open commons.
[Benkler, et al., 2016; Benkler, 2015]

The term that academics use to describe the new mode of production is "peer production."

The term was coined by Yochai Benkler as part of a theory that argues that new information tech has driven the transaction costs associated with collective action (that is, the extra costs over-and-above the act of actually doing the task) so low as to allow the mass aggregating many small contributions. The result, B argues, is a new form of collaborative production that lies outside of markets and firms.

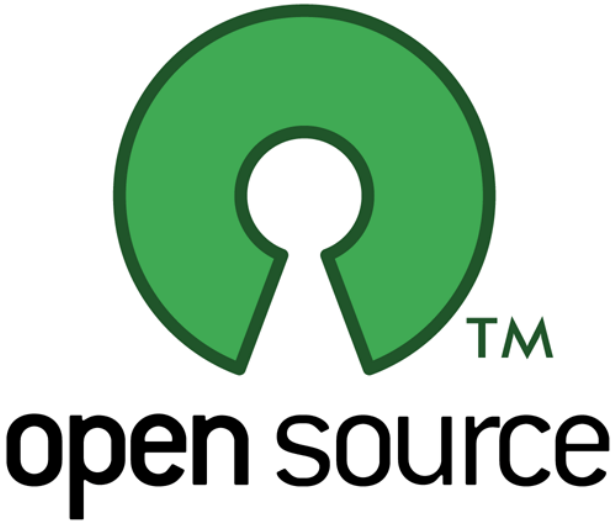
The idea was that if we published opens we would (sometimes) attract communities.

And if we did attract a community, it would improve our software.

The result was high quality and, it turns out, profits for the folks doing it.

And although it didn't always work, it sometimes did. You must work openly to build a community. Once you build a community, you really do benefit.

And if this is all sounding familiar, it should. Because a group of people who had been participating the broader free software movement started making exactly this argument in the late 1990s.



"They that should not be named at LibrePlanet."

Seriously though. Peer production lies at the heart of the argument for open source. The mission statement of the Open Source Initiative lays this out clearly.

"Open source enables a development method for software that harnesses the power of distributed peer review and transparency of process. The promise of open source is higher quality, better reliability, greater flexibility, lower cost, and an end to predatory vendor lock-in."

—Open Source Initiative mission statement

"Open source enables a development method for software that harnesses the power of distributed peer review and transparency of process. The promise of open source is higher quality, better reliability, greater flexibility, lower cost, and an end to predatory vendor lock-in."

And nearly all of the big success stories of free software are driven by peer production.

This was on the t-shirt for LibrePlanet 2010. These are the poster children of the free software world and their success is due to the kind of massive of collaboration, self-direction and self-governance, and collaborative and social production.

And free software inspired the world.

And it wasn't just software and encyclopedias!

WikiHow, OpenStreetMap, P2PU. You can look at the program for Libre-Planet and get a snapshot of a fraction of the kinds of examples. And it's incredible and inspiring.

Yesterday Public Lab joined a long list of organizations that include Wikipedia and Creative Commons and others in receiving the FSF's social benefit award that goes to organizations that have extended and applied free software principles to other areas.

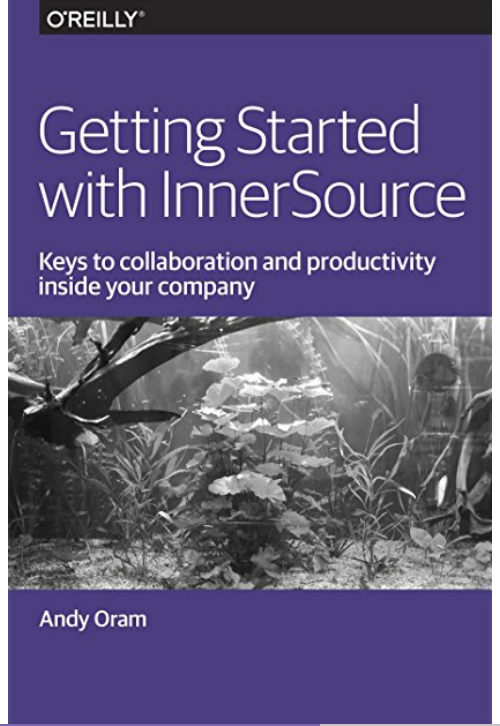
There are literally thousands of other examples.

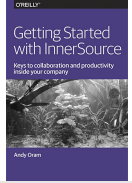
A history of online cooperative production in three “eras”

Era III: Markets learn from peer production

I want to suggest that markets players, and many people who had no interest in freedom at all were inspired by peer production as well.

And our movement is still coming to terms with the results.





In the simplest form, large firms have used "inner sourcing" to adapt technological and social infrastructure (think version control, open bug tracking, patch review—all the tricks that make peer production work) to create software that is "free" (but only distributed within a company) or proprietary software that is developed openly within the boundaries of firms.

It tends to work well in very large firms where you can build subcommunities within them. Think HP or IBM.

Andy Oram, the author of this book has been coming to LibrePlanet here earlier and deeply understands free software and is committed to our project. But understands it well enough to understand that the things we do can work for companies when they're not building free software as well.

Strategic openness.

Strategic openness.

The broader and more important change has been the development of I’ve heard other people call “strategic openness” which is companies carefully designing systems that allow for mass collaboration, but that tightly find ways to tightly control it.

Strategic closedness.

Strategic closedness.

Because strategical openness is really, fudementally, more about strategic closedness.

The more powerful example of strategic openness is Apple App Store.

The App Store, you may remember, was not part of Apple’s original vision for the iPhone when it was released in 2007.

Apple had created a number of applications that iPhones came with; users were limited to those applications.

Frustrated by this, hackers found ways to hack or “jailbreak” their iPhones. A major reason (not the only reason) was to be able to install arbitrary applications. At one point, analysts estimated that a minimum of 25% of iPhones had been “jailbroken.”

In this early period, there were many unauthorized applications had been written for jailbroken iPhones. Many were actually free software!

Apple didn’t like this but they also saw the power this “forced-open” platform.

After a extended game of threats, and attacks, Apple decided to allow users to continue using apps, but to require that they go through an Apple-controlled gatekeeper which they called the “App Store.”

The business-brilliance of the App Store is that effectively decentralized production (anybody can create apps) while placing Apple in a uniquely powerful position to capture value and control users.

Strategic openness. Strategic closedness.

Over 2 million apps in the Apple App Store have been downloaded over 130 billion times. The App Store is not free software. It’s not peer production. But it supports a software ecosystem whose value flows from the mass aggregation of many small contribution of people who, in large part, are doing it outside their jobs and it is like peer production in critical ways.

The App Store model, more than anything that came before in software, was a discovery of how big companies could benefit from something that looked like peer production or open source without having to get hands covered in freedom.

Peer production has matured.

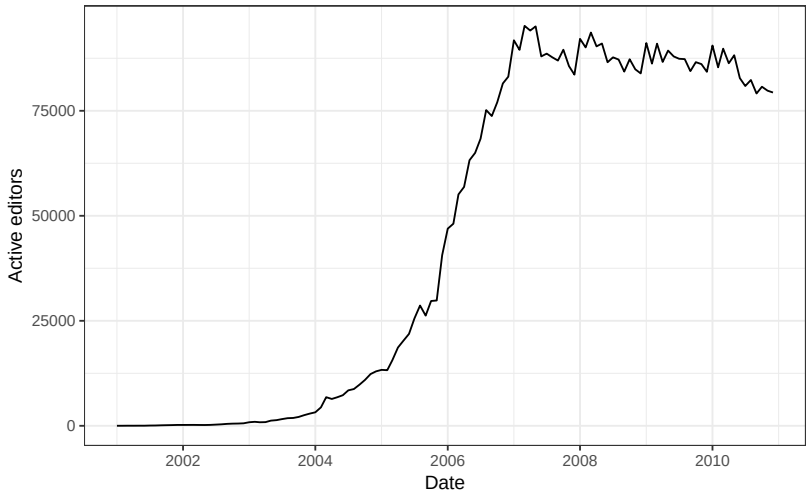
Peer production has matured.

So the world has learned from us. But we’ve also changed ourselves.

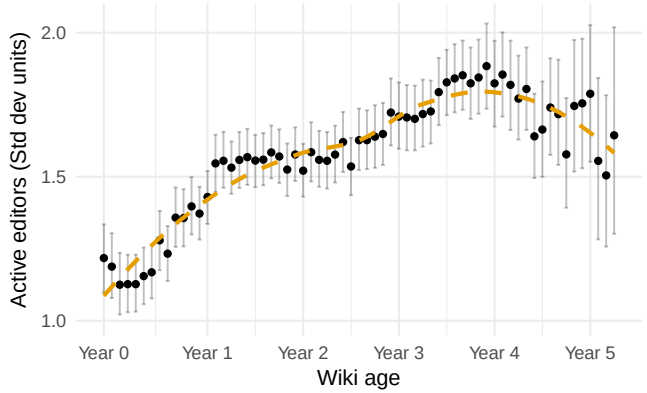
We’ve also changed ourselves.

“Mature” is a polite way to say that peer production has exited a period of meteoric growth.

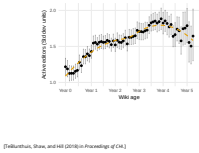
This has happened both within our most important projects and within peer production as a field.



First, we can look within projects. This is Wikipedia.



[TeBlunthuis, Shaw, and Hill (2018) in *Proceedings of CHI*.]



And this is absolutely a broader pattern.

[Explain the slide.]

But life-cycles happen to proprietary tools tool. The bigger concern is that peer production has matured as a field as well.

We are simply not adding new successful cases the way were 15 years ago.

The most successful peer production commons nearly all pre-date the iPhone.

- Linux (1991)
- Apache (1995)
- Wikipedia (2001)
- OpenStreetMap (2004)
- StackExchange (2009)

The most successful peer production commons nearly all pre-date the iPhone.
- Linux (1991)
- Apache (1995)
- Wikipedia (2001)
- OpenStreetMap (2004)
- StackExchange (2009)

This is the claim I'm least sure about in this talk but I believe that peer production's poster children were nearly all created more than 10 years ago.

- Linux (1991)
- Apache (1995)
- Wikipedia (2001)
- OpenStreetMap (2004)
- StackExchange (2009)

I know some of you can think of exceptions. So can I. And it's the nature of these exceptions that has me even more worried.

- Software for phones and embedded systems (e.g., Android)
- Cloud services (e.g., OpenStack, Kubernetes, databases, web frameworks, etc)

- Software for phones and embedded systems (e.g., Android)
- Cloud services (e.g., OpenStack, Kubernetes, databases, web frameworks, etc)

- Software for phones and embedded systems (e.g., Android)
- Cloud services (e.g., OpenStack, Kubernetes, databases, web frameworks, etc)

The firms developing and deploying get freedom. The end-users do not.

The firms developing and deploying get freedom. The end-users do not.

Although these are free software and peer production they are driven by companies engaging in strategic openness so that:

> *Companies developing and deploying the software get freedom. Users do not.*

This happens in three distinct ways:

The firms developing and deploying get freedom. The end-users do not.

- Non-copyleft licenses mean that users don't get source.
- Locked-down devices mean users cannot modify their their software (e.g., DRM, Tivoization)
- Software runs on others' computers (e.g., SaaS)

The firms developing and deploying get freedom. The end-users do not.
- Non-copyleft licenses mean that users don't get source.
- Locked-down devices mean users cannot modify their their software (e.g., DRM, Tivoization)
- Software runs on others' computers (e.g., SaaS)

- Non-copyleft licenses mean that users don't get source.
- Locked-down devices mean users cannot modify their their software (e.g., DRM, Tivoization)
- Software runs on others' computers (e.g., SaaS)

And it is absolutely better than Android and OpenStack are free software than that they were proprietary—for the sake of the SaaS companies and the phone manufacturers who have all the freedoms we care about!

But need to recognize that the users of both systems are almost all unfree. Android and OpenStack are not unambigous victories for software freedom, at least not for most users. They are examples of how free licensing no longer signifies the type of victory it might have two decades ago.

This is very bad news.

While we have matured as a movement, our opponents have found ways to benefit from the kind of mass collaboration that used to be something that only we could do.

Now they can do it too. And they're really good at it. And they're using to basically work against our goals.

Or they're doing it in ways that benefit and provide freedom to themselves but not users.

Whither Peer Production?

Good News

So where does that leave us?

Well, there's good news too!

Peer production projects remain strong.

Peer production projects remain strong.

Peer production is not dead.

Wikipedia is still the 5th most popular website in the world.

OSM and StackOverflow are thriving.

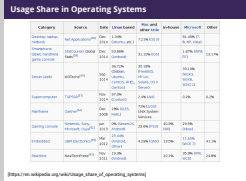
Everyone of our friends from that 2010 t-shirt is alive and doing well.

Peer productions poster children are, for the most part, all doing great.

Usage Share in Operating Systems

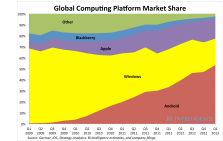
Category	Source	Date	Linux based	Mac and other Unix	In-house	Microsoft	Other
Desktop, laptop, netbook	Net Applications[58]	Dec 2014	1.34% (Ubuntu, etc.)	7.21% (OS X)		91.45% (7, 8, XP, Vista)	
Smartphone, tablet, handheld game console	StatCounter Global Stats[59]	Dec 2014	53.86% (Android)	31.10% (iOS)		1.87% (WP8, RT)	13.17%
Server (web)	W3Techs[60]	Sep 2014	36.72% (Debian, Ubuntu, CentOS, RHEL, Gentoo)	30.18% (FreeBSD, HP-UX, Solaris, OS X Server)		33.10% (W2K3, W2K8, W2K12)	
Supercomputer	TOP500[57]	Nov 2014	97.0% (Custom)	2.4% (AIX)		0.2%	0.2%
Mainframe	Gartner[54]	Dec 2008	28% (SLES, RHEL)	72% (z/OS) UNIX System Services			
Gaming console	Nintendo, Sony, Microsoft, Ouya[61]	Jun 2013	0% (SteamOS, Android)	29.6% (PS3)	40.9% (Wii)	29.5% (Xbox)	
Embedded	UBM Electronics[62]	Mar 2012	29.44% (Android, Other)	4.29% (QNX)	13.5%	11.65% (WCE 7)	41.1%
Real time	NewTechPress[63]	Nov 2011	19.3% (Android)		20.1%	35.8% (XPE, WCE)	24.8%

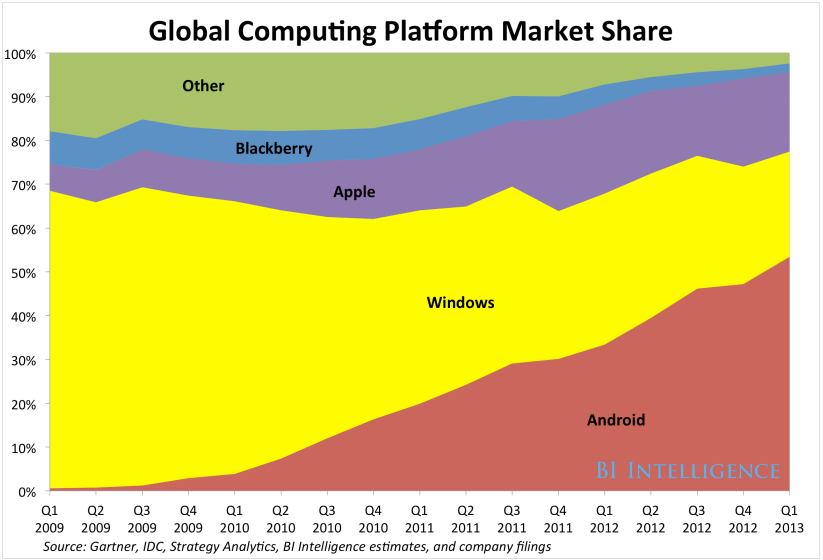
[https://en.wikipedia.org/wiki/Usage_share_of_operating_systems]



Free software runs:

- nearly 100% of supercomputers
- a large majority of severs
- most smart phones and a growing number of tablets and embedded systems





Because of the last point, a majority of computers in the world

Most valuable online informations commons are not collaborative.

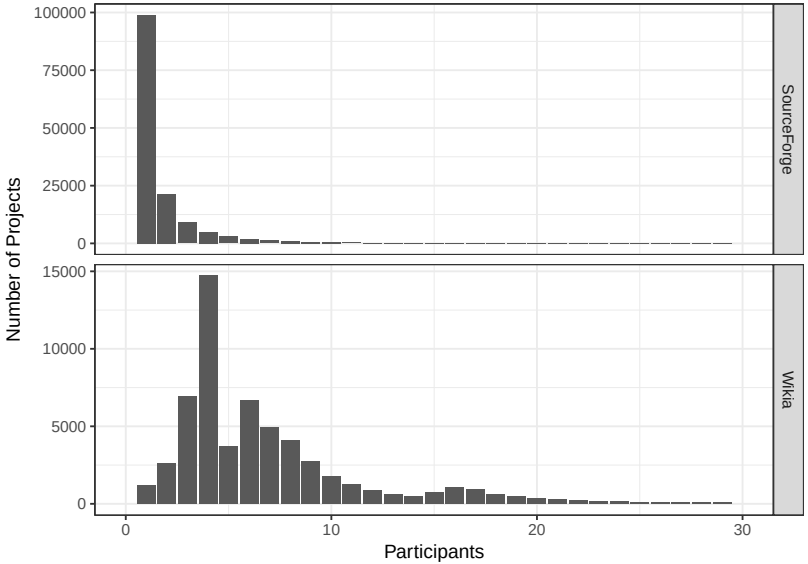
Most valuable online informations commons are not collaborative.

And even if peer production were dead, that would also be OK.

But although most of free software movements greatest successes are examples of peer production, most free software projects are not!

At a previous talk I gave at LibrePlanet, I played a game where I asked people to guess the medium number of contributors to a free software project.

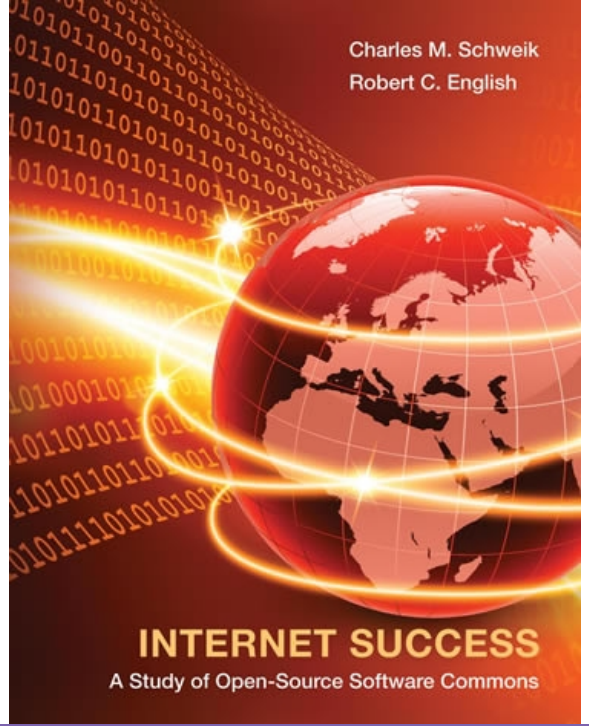
We'll skip the game this time.

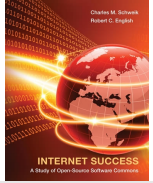


The median is 1. A single person. The top graph is data taken from SourceForge but you see similar patterns with data from GitHub, GitLab installations etc.

The bottom graph is wikis from Wikia where the median is 5, several of which are typically bots.

The vast majority of free software projects. The vast majority of wikis. The vast majority of information commons online—even successful ones we all rely on—are not peer production because they're not collaborative!





And this really is still good news!

In their excellent book Internet Success, Charlie Schweik and Bob English show that most free software projects that are downloaded, release repeatedly, and provide value to the world are entirely uncollaborative.

We don't need peer production to create real value.

Whither Peer Production?

Where do we go from here?

Whither Peer Production?
Where do we go from here?

Despite this good news, and despite our positions of strength, the challenges that I layed out are enormous.

The implications of new forms of strategic openness means that many of our greatest achievements, of late, come with big asterisks.

For example, free software's rapid dissemination through Android has come at the expense of the fact that the practical impact of freedom for most of our users is, if not zero, much lower than we would want.

Commons will typically not be able to rely on a competitive advantage against proprietary modes of production.

Commons will typically not be able to rely on a competitive advantage against proprietary modes of production.

What I'm suggesting today is something I've not heard others in our movement point us but which we must know and plan for:

The opponents of software freedom have successfully co-opted our most effective weapon (cooperative production) and they are using it to more effectively subjugate users.

They haven't completely co-opted it. And they will probably never be able to co-opt peer production *completely*. But mostly. And in many places, like the example of Airbnb, they are more effective with that particular weapon than we will likely ever be.

In the very best case, we've got to come to terms with the fact that our opponents now have the power of online distributed collaborative production too.

Commons will typically not be able to rely on a competitive advantage against proprietary modes of production.

Commons will typically not be able to rely on a competitive advantage against proprietary modes of production.

The upside, I suppose, is that thing are no worse then were in 1990.

We fought then. And we will fight now.

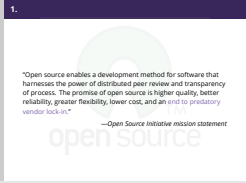
The most important implication, is that, given this new reality, we need to act differently going forward.

I've come up with five implications, as as a first stab, that I think we should consider.

1.

"Open source enables a development method for software that harnesses the power of distributed peer review and transparency of process. The promise of open source is higher quality, better reliability, greater flexibility, lower cost, and an end to predatory vendor lock-in."

—Open Source Initiative mission statement



Our advocacy needs to focus on inherent benefits of working in freedom. We need to remind people what "strategically closed" in their "strategically open" systems.

The end to predatory vendor lock-in bit of the open source mission is, in fact, a benefit of freedom. We need to refocus on advocacy built around the benefits of living in freedom.

I am absolutely not suggesting we talk about things in terms of open source, for many reasons. But we can build on the one thing they did get right, which as it turns out, was something we've been saying the whole time.

2.

Advocacy focused on reaching end users not companies.

2.
Advocacy focused on reaching end users not companies.

The target of our advocacy should shift from firms, who have driven our greatest success in the past, to users. This marks an enormous challenge because we have historically not been particularly good at reaching them.

Deb Nicholson's opening talk about broadening participation touched on these issues in depth and we need to listen.

3.

We should learn from groups fighting for other public goods like the environment, public infrastructure, public broadcasting, etc.

3.
We should learn from groups fighting for other public goods like the environment, public infrastructure, public broadcasting, etc.

Perhaps the most thing we can do is learn from groups fighting for other public goods like the environment, public infrastructure, public broadcasting, etc.

This will rely more on traditional strategies for providing public goods.

4.

Working with government and engaging in lobbying.

4.
Working with government and engaging in lobbying.

Working with government and engaging in lobbying.

In terms of government: Many state actors are already actively engaged in this process. In 2017, both the city government of Barcelona and the state government of of Kerala (a province in India) mandated the use and development of FLOSS in government and adjacent industry.

RMS mentioned a half dozen more including even some important inroads in the US. We need to build on these successes.

5.

Supporting change through civil-society organizations, non-profits, and activism.

5.
Supporting change through civil-society organizations, non-profits, and activism.

Supporting change through civil-society organizations, non-profits, and activism.

This might involve new broader coalitions. For example, groups working to promote commons in general may act as natural allies.

Mostly, it will involve supporting the non-profits already active in this space.

The FSF is perhaps the most notable and most important. We need to support the FSF now more than we have at any point in the last 20 years.

Two decades ago, due to the hard work of our movement, we put ourselves in a position—truly an incredible position—where people who did not share our values were achieving victories for our ideals!

If what I said today is right, we can't count on this as much in the future.

Our most effective allies, who were never really shared our goals anyway and who RMS was always warning us about, will not be carrying our fight with us in the future.

More than at any point in the recent past, the success of the free software movement depends on the work of people who are committed to software freedom.

More than at any point in the recent past, the success of the free software movement depends on the work of people who are committed to software freedom.

What that means is that, more than at any point in the last 25 years, the success of our movement depends on the work of people who are committed to software freedom.

We need to remain focused on the fact as technology permeates and mediates every aspect of our lives, the political importance of control over our technology has not decreased. Indeed, it is more important today than ever.

But I believe that we can longer count on large companies to fix it for us. We must lead the charge ourselves.

I look forward to joining you all in the struggle.

Thanks!

Continue this conversation!

email: mako@atdot.cc

mastodon: mako@social.coop

Thanks!
Continue this conversation!
email: mako@atdot.cc
mastodon: mako@social.coop