

# What Business Can Learn from Open Source

paulgraham.com · Paul Graham · 2005-08 · [source](#)

---

|

August 2005

(This essay is derived from a talk at Oscon 2005.)

Lately companies have been paying more attention to open source. Ten years ago there seemed a real danger Microsoft would extend its monopoly to servers. It seems safe to say now that open source has prevented that. A recent survey found 52% of companies are replacing Windows servers with Linux servers.

[1]

More significant, I think, is which 52% they are. At this point, anyone proposing to run Windows on servers should be prepared to explain what they know about servers that Google, Yahoo, and Amazon don't.

But the biggest thing business has to learn from open source is not about Linux or Firefox, but about the forces that produced them. Ultimately these will affect a lot more than what software you use.

We may be able to get a fix on these underlying forces by triangulating from open source and blogging. As you've probably noticed, they have a lot in common.

Like open source, blogging is something people do themselves, for free, because they enjoy it. Like open source hackers, bloggers compete with people working for money, and often win. The method of ensuring quality is also the same: Darwinian. Companies ensure quality through rules to prevent employees from screwing up. But you don't need that when the audience can communicate with one another. People just produce whatever they want; the good stuff spreads, and the bad gets ignored. And in both cases, feedback from the audience improves the best work.

Another thing blogging and open source have in common is the Web. People have always been willing to do great work for free, but before the Web it was harder to reach an audience or collaborate on projects.

## Amateurs

I think the most important of the new principles business has to learn is that people work a lot harder on stuff they like. Well, that's news to no one. So how can I claim business has to learn it? When I say business doesn't know this, I mean the structure of business doesn't reflect it.

Business still reflects an older model, exemplified by the French word for working: *travailler*. It has an English cousin, *travail*, and what it means is torture.

[2]

This turns out not to be the last word on work, however. As societies get richer, they learn something about work that's a lot like what they learn about diet. We know now that the healthiest diet is the one our peasant ancestors were forced to eat because they were poor. Like rich food, idleness only seems desirable when you don't get enough of it. I think we were designed to work, just as we were designed to eat a certain amount of fiber, and we feel bad if we don't.

There's a name for people who work for the love of it: amateurs. The word now has such bad connotations that we forget its etymology, though it's staring us in the face. "Amateur" was originally rather a complimentary word. But the thing to be in the twentieth century was professional, which amateurs, by definition, are not.

That's why the business world was so surprised by one lesson from open source: that people working for love often surpass those working for money. Users don't switch from Explorer to Firefox because they want to hack the source. They switch because it's a better browser.

It's not that Microsoft isn't trying. They know controlling the browser is one of the keys to retaining their monopoly. The problem is the same they face in operating systems: they can't pay people enough to build something better than a group of inspired hackers will build for free.

I suspect professionalism was always overrated-- not just in the literal sense of working for money, but also connotations like formality and detachment. Inconceivable as it would have seemed in, say, 1970, I think professionalism was largely a fashion, driven by conditions that happened to exist in the twentieth century.

One of the most powerful of those was the existence of "channels." Revealingly, the same term was used for both products and information: there were distribution channels, and TV and radio channels.

It was the narrowness of such channels that made professionals seem so superior to amateurs. There were only a few jobs as professional journalists, for example, so competition ensured the average journalist was fairly good. Whereas anyone can express opinions about current events in a bar. And so the average person expressing his opinions in a bar sounds like an idiot compared to a journalist writing about the subject.

On the Web, the barrier for publishing your ideas is even lower. You don't have to buy a drink, and they even let kids in. Millions of people are publishing online, and the average level of what they're writing, as you might expect, is not very good. This has led some in the media to conclude that blogs don't present much of a threat-- that blogs are just a fad.

Actually, the fad is the word "blog," at least the way the print media now use it. What they mean by "blogger" is not someone who publishes in a weblog format, but anyone who publishes online. That's going to become a problem as the Web becomes the default medium for publication. So I'd like to suggest an alternative word for someone who publishes online. How about "writer?"

Those in the print media who dismiss the writing online because of

its low average quality are missing an important point: no one reads the average blog. In the old world of channels, it meant something to talk about average quality, because that's what you were getting whether you liked it or not.

But now you can read any writer you want. So the average quality of writing online isn't what the print media are competing against. They're competing against the best writing online. And, like Microsoft, they're losing.

I know that from my own experience as a reader. Though most print publications are online, I probably read two or three articles on individual people's sites for every one I read on the site of a newspaper or magazine.

And when I read, say, New York Times stories, I never reach them through the Times front page. Most I find through aggregators like Google News or Slashdot or Delicious. Aggregators show how much better you can do than the channel. The New York Times front page is a list of articles written by people who work for the New York Times. Delicious is a list of articles that are interesting. And it's only now that you can see the two side by side that you notice how little overlap there is.

Most articles in the print media are boring. For example, the president notices that a majority of voters now think invading Iraq was a mistake, so he makes an address to the nation to drum up support. Where is the man bites dog in that? I didn't hear the speech, but I could probably tell you exactly what he said. A speech like that is, in the most literal sense, not news: there is nothing new in it.

[3]

Nor is there anything new, except the names and places, in most "news" about things going wrong. A child is abducted; there's a tornado; a ferry sinks; someone gets bitten by a shark; a small plane crashes. And what do you learn about the world from these stories? Absolutely nothing. They're outlying data points; what makes them gripping also makes them irrelevant.

As in software, when professionals produce such crap, it's not

surprising if amateurs can do better. Live by the channel, die by the channel: if you depend on an oligopoly, you sink into bad habits that are hard to overcome when you suddenly get competition.

[4]

## Workplaces

Another thing blogs and open source software have in common is that they're often made by people working at home. That may not seem surprising. But it should be. It's the architectural equivalent of a home-made aircraft shooting down an F-18. Companies spend millions to build office buildings for a single purpose: to be a place to work. And yet people working in their own homes, which aren't even designed to be workplaces, end up being more productive.

This proves something a lot of us have suspected. The average office is a miserable place to get work done. And a lot of what makes offices bad are the very qualities we associate with professionalism. The sterility of offices is supposed to suggest efficiency. But suggesting efficiency is a different thing from actually being efficient.

The atmosphere of the average workplace is to productivity what flames painted on the side of a car are to speed. And it's not just the way offices look that's bleak. The way people act is just as bad.

Things are different in a startup. Often as not a startup begins in an apartment. Instead of matching beige cubicles they have an assortment of furniture they bought used. They work odd hours, wearing the most casual of clothing. They look at whatever they want online without worrying whether it's "work safe." The cheery, bland language of the office is replaced by wicked humor. And you know what? The company at this stage is probably the most productive it's ever going to be.

Maybe it's not a coincidence. Maybe some aspects of professionalism are actually a net lose.

To me the most demoralizing aspect of the traditional office is that you're supposed to be there at certain times. There are usually a few people in a company who really have to, but the reason most employees work fixed hours is that the company can't measure their productivity.

The basic idea behind office hours is that if you can't make people work, you can at least prevent them from having fun. If employees have to be in the building a certain number of hours a day, and are forbidden to do non-work things while there, then they must be working. In theory. In practice they spend a lot of their time in a no-man's land, where they're neither working nor having fun.

If you could measure how much work people did, many companies wouldn't need any fixed workday. You could just say: this is what you have to do. Do it whenever you like, wherever you like. If your work requires you to talk to other people in the company, then you may need to be here a certain amount. Otherwise we don't care.

That may seem utopian, but it's what we told people who came to work for our company. There were no fixed office hours. I never showed up before 11 in the morning. But we weren't saying this to be benevolent. We were saying: if you work here we expect you to get a lot done. Don't try to fool us just by being here a lot.

The problem with the facetime model is not just that it's demoralizing, but that the people pretending to work interrupt the ones actually working. I'm convinced the facetime model is the main reason large organizations have so many meetings. Per capita, large organizations accomplish very little. And yet all those people have to be on site at least eight hours a day. When so much time goes in one end and so little achievement comes out the other, something has to give. And meetings are the main mechanism for taking up the slack.

For one year I worked at a regular nine to five job, and I remember well the strange, cozy feeling that comes over one during meetings. I was very aware, because of the novelty, that I was being paid for programming. It seemed just amazing, as if there was a machine on my desk that spat out a dollar bill every two minutes no matter

what I did. Even while I was in the bathroom! But because the imaginary machine was always running, I felt I always ought to be working. And so meetings felt wonderfully relaxing. They counted as work, just like programming, but they were so much easier. All you had to do was sit and look attentive.

Meetings are like an opiate with a network effect. So is email, on a smaller scale. And in addition to the direct cost in time, there's the cost in fragmentation-- breaking people's day up into bits too small to be useful.

You can see how dependent you've become on something by removing it suddenly. So for big companies I propose the following experiment. Set aside one day where meetings are forbidden-- where everyone has to sit at their desk all day and work without interruption on things they can do without talking to anyone else.

Some amount of communication is necessary in most jobs, but I'm sure many employees could find eight hours worth of stuff they could do by themselves. You could call it "Work Day."

The other problem with pretend work is that it often looks better than real work. When I'm writing or hacking I spend as much time just thinking as I do actually typing. Half the time I'm sitting drinking a cup of tea, or walking around the neighborhood. This is a critical phase-- this is where ideas come from-- and yet I'd feel guilty doing this in most offices, with everyone else looking busy.

It's hard to see how bad some practice is till you have something to compare it to. And that's one reason open source, and even blogging in some cases, are so important. They show us what real work looks like.

We're funding eight new startups at the moment. A friend asked what they were doing for office space, and seemed surprised when I said we expected them to work out of whatever apartments they found to live in. But we didn't propose that to save money. We did it because we want their software to be good. Working in crappy informal spaces is one of the things startups do right without realizing it. As soon as you get into an office, work and life start to drift apart.

That is one of the key tenets of professionalism. Work and life are supposed to be separate. But that part, I'm convinced, is a mistake.

## Bottom-Up

The third big lesson we can learn from open source and blogging is that ideas can bubble up from the bottom, instead of flowing down from the top. Open source and blogging both work bottom-up: people make what they want, and the best stuff prevails.

Does this sound familiar? It's the principle of a market economy. Ironically, though open source and blogs are done for free, those worlds resemble market economies, while most companies, for all their talk about the value of free markets, are run internally like communist states.

There are two forces that together steer design: ideas about what to do next, and the enforcement of quality. In the channel era, both flowed down from the top. For example, newspaper editors assigned stories to reporters, then edited what they wrote.

Open source and blogging show us things don't have to work that way. Ideas and even the enforcement of quality can flow bottom-up. And in both cases the results are not merely acceptable, but better. For example, open source software is more reliable precisely because it's open source; anyone can find mistakes.

The same happens with writing. As we got close to publication, I found I was very worried about the essays in Hackers & Painters that hadn't been online. Once an essay has had a couple thousand page views I feel reasonably confident about it. But these had had literally orders of magnitude less scrutiny. It felt like releasing software without testing it.

That's what all publishing used to be like. If

you got ten people to read a manuscript, you were lucky. But I'd become so used to publishing online that the old method now seemed alarmingly unreliable, like navigating by dead reckoning once you'd gotten used to a GPS.

The other thing I like about publishing online is that you can write what you want and publish when you want. Earlier this year I wrote something that seemed suitable for a magazine, so I sent it to an editor I know.

As I was waiting to hear back, I found to my surprise that I was hoping they'd reject it. Then I could put it online right away. If they accepted it, it wouldn't be read by anyone for months, and in the meantime I'd have to fight word-by-word to save it from being mangled by some twenty five year old copy editor.

[5]

Many employees would like to build great things for the companies they work for, but more often than not management won't let them. How many of us have heard stories of employees going to management and saying, please let us build this thing to make money for you-- and the company saying no? The most famous example is probably Steve Wozniak, who originally wanted to build microcomputers for his then-employer, HP. And they turned him down. On the blunderometer, this episode ranks with IBM accepting a non-exclusive license for DOS. But I think this happens all the time. We just don't hear about it usually, because to prove yourself right you have to quit and start your own company, like Wozniak did.

## Startups

So these, I think, are the three big lessons open source and blogging have to teach business: (1) that people work harder on stuff they like, (2) that the standard office environment is very unproductive, and (3) that bottom-up often works better than top-down.

I can imagine managers at this point saying: what is this guy talking about? What good does it do me to know that my programmers would be more productive working at home on their own projects? I need their asses in here working on version 3.2 of our software, or we're never going to

make the release date.

And it's true, the benefit that specific manager could derive from the forces I've described is near zero. When I say business can learn from open source, I don't mean any specific business can. I mean business can learn about new conditions the same way a gene pool does. I'm not claiming companies can get smarter, just that dumb ones will die.

So what will business look like when it has assimilated the lessons of open source and blogging? I think the big obstacle preventing us from seeing the future of business is the assumption that people working for you have to be employees. But think about what's going on underneath: the company has some money, and they pay it to the employee in the hope that he'll make something worth more than they paid him. Well, there are other ways to arrange that relationship. Instead of paying the guy money as a salary, why not give it to him as investment? Then instead of coming to your office to work on your projects, he can work wherever he wants on projects of his own.

Because few of us know any alternative, we have no idea how much better we could do than the traditional employer-employee relationship. Such customs evolve with glacial slowness. Our employer-employee relationship still retains a big chunk of master-servant DNA.

[6]

I dislike being on either end of it. I'll work my ass off for a customer, but I resent being told what to do by a boss. And being a boss is also horribly frustrating; half the time it's easier just to do stuff yourself than to get someone else to do it for you. I'd rather do almost anything than give or receive a performance review.

On top of its unpromising origins, employment has accumulated a lot of cruft over the years. The list of what you can't ask in job interviews is now so long that for convenience I assume it's infinite. Within the office you now have to walk on eggshells lest anyone

say or do

something that makes the company prey to a lawsuit. And God help you if you fire anyone.

Nothing shows more clearly that employment is not an ordinary economic relationship than companies being sued for firing people. In any purely economic relationship you're free to do what you want. If you want to stop buying steel pipe from one supplier and start buying it from another, you don't have to explain why. No one can accuse you of unjustly switching pipe suppliers. Justice implies some kind of paternal obligation that isn't there in transactions between equals.

Most of the legal restrictions on employers are intended to protect employees. But you can't have action without an equal and opposite reaction. You can't expect employers to have some kind of paternal responsibility toward employees without putting employees in the position of children. And that seems a bad road to go down.

Next time you're in a moderately large city, drop by the main post office and watch the body language of the people working there. They have the same sullen resentment as children made to do something they don't want to. Their union has exacted pay increases and work restrictions that would have been the envy of previous generations of postal workers, and yet they don't seem any happier for it. It's demoralizing to be on the receiving end of a paternalistic relationship, no matter how cozy the terms. Just ask any teenager.

I see the disadvantages of the employer-employee relationship because I've been on both sides of a better one: the investor-founder relationship. I wouldn't claim it's painless. When I was running a startup, the thought of our investors used to keep me up at night. And now that I'm an investor, the thought of our startups keeps me up at night. All the pain of whatever problem you're trying to solve is still there. But the pain hurts less when it isn't mixed with resentment.

I had the misfortune to participate in what amounted to a controlled experiment to prove that. After Yahoo bought our startup I went to work for them. I was doing exactly the same work, except with bosses. And to my horror I started acting like a child. The situation pushed buttons I'd forgotten I had.

The big advantage of investment over employment, as the examples of open source and blogging suggest, is that people working on projects of their own are enormously more productive. And a startup is a project of one's own in two senses, both of them important: it's creatively one's own, and also economically one's own.

Google is a rare example of a big company in tune with the forces I've described. They've tried hard to make their offices less sterile than the usual cube farm. They give employees who do great work large grants of stock to simulate the rewards of a startup. They even let hackers spend 20% of their time on their own projects.

Why not let people spend 100% of their time on their own projects, and instead of trying to approximate the value of what they create, give them the actual market value? Impossible? That is in fact what venture capitalists do.

So am I claiming that no one is going to be an employee anymore-- that everyone should go and start a startup? Of course not.

But more people could do it than do it now.

At the moment, even the smartest students leave school thinking they have to get a job.

Actually what they need to do is make something valuable. A job is one way to do that, but the more ambitious ones will ordinarily be better off taking money from an investor than an employer.

Hackers tend to think business is for MBAs. But business administration is not what you're doing in a startup. What you're doing is business creation. And the first phase of that is mostly product creation-- that is, hacking. That's the hard part. It's a lot harder to create something people love than

to take something people love and figure out how to make money from it.

Another thing that keeps people away from starting startups is the risk. Someone with kids and a mortgage should think twice before doing it. But most young hackers have neither.

And as the example of open source and blogging suggests, you'll enjoy it more, even if you fail. You'll be working on your own thing, instead of going to some office and doing what you're told. There may be more pain in your own company, but it won't hurt as much.

That may be the greatest effect, in the long run, of the forces underlying open source and blogging: finally ditching the old paternalistic employer-employee relationship, and replacing it with a purely economic one, between equals.

## Notes

[1]

Survey by Forrester Research reported in the cover story of Business Week, 31 Jan 2005. Apparently someone believed you have to replace the actual server in order to switch the operating system.

[2]

It derives from the late Latin *tripalium*, a torture device so called because it consisted of three stakes. I don't know how the stakes were used. "Travel" has the same root.

[3]

It would be much bigger news, in that sense, if the president faced unscripted questions by giving a press conference.

[4]

One measure of the incompetence of newspapers is that so many still make you register to read stories. I have yet to find a blog that tried that.

[5]

They accepted the article, but I took so long to send them the final version that by the time I did the section of the magazine they'd accepted it for had disappeared in a reorganization.

[6]

The word "boss" is derived from the Dutch baas, meaning "master."

Thanks to Sarah Harlin, Jessica Livingston, and Robert Morris for reading drafts of this.

|