Open Source Initiative

Open Source Case for Business:Advocacy

Sat, 2007-03-31 06:03 — nelson

Open Source Case for Business

Supportive Documents:

- Ernie Ball (guitar string manufacturer) switches to open source and saves $80,000.
- Open Source-onomics: Examining some pseudo-economic arguments about Open Source.
- MITRE REPORT: "A Business Case Study of Open Source Software".
- "Your Open Source Plan" from CIO magazine.

The open-source model has a lot to offer the business world. It's a way to build open standards as actual software, rather than paper documents. It's a way that many companies and individuals can collaborate on a product that none of them could achieve alone. It's the rapid bug-fixes and the changes that the user asks for, done to the user's own schedule.

The open-source model also means increased security; because code is in the public view it will be exposed to extreme scrutiny, with problems being found and fixed instead of being kept secret until the wrong person discovers them. And last but not least, it's a way that the little guys can get together and have a good chance at beating a monopoly.

Of all these benefits, the most fundamental is increased reliability. And if that's too abstract for you, you should think about how closed sources made the Year 2000 problem worse and why they might have very well killed your business.

The Reliability Problem

Gerald P. Weinberg once famously observed that, "If builders built houses the way programmers built programs, the first woodpecker to come along would destroy civilization." He was right. Up to now, the reliability of most software has been atrociously bad.

The foundation of the business case for open-source is high reliability. Open-source software is peer-reviewed software; it is more reliable than closed, proprietary software. Mature open-source code is as bulletproof as software ever gets.

Until recently this was a radical idea to many businesspeople; many had a belief that open-source software is necessarily not "professional," that it is shoddily made and more prone to fail than closed software.

But the Internet's infrastructure makes the best possible refutation, and since OSI was founded in 1998 many people have been paying attention. Consider DNS, sendmail, the various open-source TCP/IP stacks and utility suites, and the open-source scripting languages such as Perl that are behind most "live" content on the Web. These are the running gears of the Internet. (Read this for a look at what would happen if they disappeared).

These open-source programs have demonstrated a level of reliability and robustness under fast-changing conditions (including a huge and rapid increase in the Internet's size) that, considered against the performance

record of even the best closed commercial software, is nothing short of astonishing.

You can read an extended technical argument for the superior reliability of general open-source software in "The Cathedral and the Bazaar". This paper was behind Netscape's pioneering decision to take its client software open-source. It describes a bazaar style of managing software development that depends on open source and leads to high reliability and quality.

The real-world evidence backs this up. In an independent head-to-head reliability test, open-source Unix systems and utilities were less fragile - crashed or hung less often - than their proprietary counterparts. .

The business implication of this technical case is clear. Eventually, bazaar-mode peer review will come to be considered a necessary condition for highest quality. In many market niches, software that has not been peer-reviewed simply won't be perceived as good enough to compete.

The Payoff for Software Producers

Bazaar-mode development seems to reverse our normal expectations about software development; more programmers are better (at least, as long as the capacity of the project leader or project core group to handle integration isn't exceeded). Even a small open-source project can muster more brains to improve a piece of software than most development shops can possibly afford.

You'll see the following gains under the open-source model whether you're producing software for internal use or for resale.

Advantage: Development Speed

It follows that commercial developers leveraging the bazaar mode should be able to grab, and keep, a substantial initiative advantage over those that don't. But there's more; the first commercial developer in a given market niche to switch to this mode may gain substantial advantages over later ones.

Why? Because the pool of talent available for bazaar recruitment is limited. The first bazaar project in a given niche is more likely to attract the best co-developers to invest time in it. Once they've invested the time, they're more likely to stick with it.

Advantage: Lower Overhead

Switching to the open-source model should also be good for a significant overhead reduction in per-project software production costs.

The open-source model allows software shops to (in effect) outsource some of their work, paying for it in values less tangible than money. (But perhaps not less economically significant; the increased speed with which an outside co-developer can have a needed bug fix will often translate into a substantial opportunity gain for that customer.)

This means smaller shops will be able to handle bigger projects.

The Payoff for Software Merchants

If you produce software for sale, you'll see two more advantages:

Advantage: Closeness to the Customer

One of the most often-repeated pieces of management advice is "Stay close to the customer." In today's fast-moving, short-product-cycle business climate it's more important than ever to do that - to understand almost as soon as they do what the customers want and be able to rapidly respond to those needs.

If you sell software, what better way to do this than by co-opting your customers' engineers to help your

development?

It's worth pointing out that the open-source, bazaar method resembles the way many successful Japanese companies have done consumer product development; get a product to market that works but is not perfect, and iterate quickly based upon customer feedback to reach the combination of features that the customers need and want. This has turned out to be especially valuable for high technology products (laptops, personal assistants, cellphones, etc) that people don't know they need, or what features they need.

Advantage: Broader Market

An important side-effect of the open-source model will be a much wider platform range for your product. Open-source authors frequently find themselves receving, for free, port changes for operating systems and environments they barely know exist and can't afford developers to support. Each such port, of course, widens the market appeal of the product.

The Payoff for Entrepreneurs

For an entrepreneur or start-up software producer, going open-source is a way to grab mind-share. The best new concept in the world won't make money unless people know it's interesting.

Whether this makes sense as a strategy depends on whether you think your main value proposition is in the software itself or in service and the expertise associated with the software. More often than one might think, the value is actually in service and integration.

This, to give one recent example, the startup Digital Creations open-sourced its flagship project Zope on the advice of its venture capitalists. The VCs projected that going open-source would actually increase the value of the company.

For full discussion see Paul Everitt's business decision essay. It makes an eloquent case.

You can also read *Wired* magazine's tour of open-source startups..

Four Ways To Win

Now for a higher-level, investor's point of view. There are at least four known business models for making money with open source:

1. **Support Sellers (otherwise known as "Give Away the Recipe, Open A Restaurant")**: In this model, you (effectively) give away the software product, but sell distribution, branding, and after-sale service. This is what (for example) Red Hat does.
2. **Loss Leader**: In this model, you give away open-source as a loss-leader and market positioner for closed software. This is what Netscape is doing.
3. **Widget Frosting**: In this model, a hardware company (for which software is a necessary adjunct but strictly a cost rather than profit center) goes open-source in order to get better drivers and interface tools cheaper. Silicon Graphics, for example, supports and ships Samba.
4. **Accessorizing**: Selling accessories - books, compatible hardware, complete systems with open-source software pre-installed. It's easy to trivialize this (open-source T-shirts, coffee mugs, Linux penguin dolls) but at least the books and hardware underly some clear successes: O'Reilly Associates, and SSC are among them.

The open-source culture's exemplars of commercial success have, so far, been service sellers or loss leaders. Nevertheless, there is good reason to believe that the clearest near-term gains in open-source will be in widget frosting.

For widget-makers (such as semiconductor or peripheral-card manufacturers), interface software is not even potentially a revenue source. Therefore the downside of moving to open source is minimal.

(Frank Hecker of Netscape proposes more models and discusses them in detail in his paper Setting Up Shop.)

There are even, as it turns out, people willing to argue that the open-source model could work well economically for hardware design.

Login or register to post comments