By Esther Schindler, ITworld
SEPTEMBER 11, 2009

Why Users Dumped Your Open Source App for Proprietary Software

Hard as it may be to imagine, "free" is not always the primary selling point.

FOSS adherents are happy to discuss all the reasons that open source is attractive to users and to other developers, from "it's free!" to "the philosophy of open source." Sometimes, they talk about the reasons that people avoid open source, such as "I want a phone number for tech support." But the hard fact is that sometimes people try an open source application — such as *yours* — and they end up not using it. I realize this is hard to imagine. But it happens, and not merely because the users have evil in their hearts.

[See also: Convincing the Boss to Accept FOSS]

While that's copacetic for plenty of people in the FOSS community (the not-adopting the software, I mean, not the evil part), other folks truly want their software to be used, and they want other developers to contribute to enhancing the app. (Whether you also want press attention to give your project better visibility is something else again.) I hate to bring up a dirty word, but the applicable term is *marketing*. That is: if you want more people to use your stuff, you need to know what it is that's currently chasing them away, and (assuming you care) you need to address those issues.

So I asked several people, especially open-source-friendly techies, about the times they seriously experimented with or used an open source app — and ended up using a proprietary application after all. I'm not speaking of a "trial" scenario in which someone downloaded an open source app, poked at it for a couple of hours, and decided, "Eh." I'm

thinking more about situations in which they (or the department, or the company) were prepared to commit to using the software (for personal or business purposes) and decided on a non-open-source option instead.

My aim is not to diminish open source; quite to the contrary. My view is that — to use ordinary marketing terms — to sell an open source app you have to understand and respond to the sales objections. An open source project (or at least those in it who are thinking, "How do we get more users?") has to understand the ways in which it might "lose to the competition" before it can address the problem.

The most common answer is the one you probably expect: **It didn't have the features required.** That's understandable, because few open source apps have as many hours of development as do mature (you can read "bloated" if you like) proprietary alternatives. Any software that's been around for 10 or 15 years is apt to have more feature depth than will a newer application.

For example, one user rejected OpenOffice.org for his Mac because its builds were "woefully inadequate." He explained, "I tried it for a while, and couldn't stomach it, so went back to the Microsoft Office that was installed on the laptop." He chose a commercial file transfer framework over as2 because the former included additional protocols and had more management capability. "It boils down to a question of maturity and scope. If the commercial application offers a great labor reduction due to management features or maturity, it becomes worth paying the extra money for," he explained.

An entrepreneur I know tried using Unison for syncing files to a central server from multiple computers. He says the performance was awful, so they went with DropBox, which works on all their OSs. (Slow performance, by the way, comes up repeatedly as a deal-killer. Given a choice between "add another feature" and "make this version faster," choose the latter.)

Sometimes it's not the feature list that's the problem, per se, but fit-and-finish. When **setup and configuration is too fiddly,** people won't take the time to mess with it. For example, one developer decided the then-current crop of open source tools were too slow and required "too much fooling around to administer" so chose Perforce, which is "just fast as

blazes." Back then, he said, "it was $750 a seat for small commercial users like us, which is kinda pricey, but we always considered it a great investment." Several answers came down to an open source app lacking polish or varying too much from familiar user interfaces; The Gimp was a repeated example.

I doubt any of those answers is surprising. But I saw a few trends, particularly between-the-lines, that are worth pointing out.

One thing that became apparent is that the lack of features is a perception that may have dated from a previous version. That is, "I tried it a few years ago, and it didn't do what I needed then, so I chose something else... and haven't thought about adopting the FOSS app since." For example, one respondent wrote, "I ditched Linux for Mac OS X, because Linux took too much tweaking, setup, and maintenance. (Note, this was before Ubuntu, which I hear is much less hassle.)" Linux missed out then, and maybe things are far different now... but we can probably assume the user is reasonably committed to the Mac now. If someone tried your app three years ago, back when it was all raw edges and bare metal, how will she know that it might be time to re-evaluate the options?

When you're inside a community (open source or otherwise), you don't see the barriers anymore, and you forgive them when you do. If you're a regular at a restaurant and the staff has one off-night, you can be patient because you know the food is worth the wait. If it's your first visit... not so much. And, importantly, you probably won't return. Initial impressions count, in software and in everything else.

One attribute of commercial releases is that major feature upgrades are announced with a lot of fanfare. That happens with open source applications that are household names (assuming an appropriately-geek household), but it's rare. Someone who tried your app three years ago and found it wanting may not realize that the version she can download today is far improved. Unless she goes out of her way to look, how likely is she to find out?

Perhaps you've gotten this far and decided my blog post is useless because it doesn't help you identify why *your* app is being rejected. Solving your individual problem is not my job. But there's nothing to keep you from asking those who download your app what they think about it. At its simplest, post a poll, asking users to rate the app (perhaps on different criteria). If you ask for an e-mail ID somewhere in the installation process, write to the

downloader two weeks later to ask if they're still using it, and, if not, why not; you can also ask what they chose instead. This can be part of the committers' setting priorities: *Is* is more important to add that feature, or to improve app performance? Perhaps it's my own market research background, but it amazes me that people just don't ask.

But perhaps the reason people dumped your open source app isn't about the software itself.

CIOs and other corporate execs are often more concerned with security and support issues, but it isn't only the bosses who put the kibosh on FOSS adoption. Several responses touched on **the desire for reliable support**, or the fear that it wasn't available for the open source app. "I guess I wanted to ensure that I had a fully supported configuration with the least number of glitches," wrote one person. Vendors obviously take advantage of this fear in their sales presentations ("If I invite in a vendor, they will do a nice PowerPoint dog-and-pony show that I can't get from the open source community," one corporate user said, cynically) and in their own willingness to *ensure* support in one way or another. "The newsletter system we had often lead us reboot Apache, this way we opted by purchasing a paid webware instead, and since we have paid for the licence, we got faster support and they managed to keep us running without Apache issues anymore," wrote another user (a little cryptically, but I think the point comes across).

Open source support can cause a lot of confusion. An accounting VAR whom I've known for 25 years explained that her company used a Microsoft open source app, an add-in to Solomon (now Microsoft Dynamics SL), for several clients. "It was the only way we could get Time Entry to work from a Macintosh," she said. "We dropped [the open source app] when the clients upgraded because we were forced to. Microsoft did not upgrade the product for the next version! *Very* sad situation; clients now had to pay for something they did not fully understand was free... messy.

Some of the people who responded to my query pooh-poohed the whole support issue because, they claim, most business-class open source apps have qualified consultants who can provide enterprise-level support. Cool. But how obvious will that be to the casual observer? Does your project web site have any list of "professionals who can help you put this WayCool technology to work"?

Politics plays a part, too. "My manager purchased the proprietary [app], and if we keep using

the open source (what we like better), it may suggest that this purchase was not necessary," admitted one person. That points to the entire issue of "how to manage your boss," which is more a management topic. Few open source projects can address that issue directly, but if you're trying to gain corporate adoption, perhaps it's worth considering creating a wiki that helps your individual users "sell it" to the company. ("Make the boss look good" is always a good idea.)

Now that you've seen why people rejected *your* open source app, it's your turn: What made you dump an open source app you were using? What could that project have done differently?