Q&A

Matt Asay

Q. Is Open Source Sustainable?

A. In 2008, many thought the open source movement could not survive the widespread adoption of open source software without commensurate contributions back, whether in code or cash. Since that time, however, open source has flourished, and it has become robustly self-sustaining.

This dramatic improvement in the health of the open-source ecosystem derives from two primary trends: a move toward more permissive, Apache-style licensing, coupled with an increase in open-source contributions from web technology companies like Facebook. Driving both trends is an increased emphasis on attracting developer communities, and not simply dollars. Perhaps surprisingly, then, the less the open-source community has focused on financial sustainability and more on developer sustainability, the more money it has made and the more sustainable it has become.

Open Source, Four Years Ago

Open source has the chance of becoming a nonrenewable resource if enterprises consume it without contributing cash or code back. Yes, there will always be open source software that doesn't rely on corporate patronage, ***but it may not be the type and caliber of code that enterprises require.*** (Asay, 2008; tinyurl.com/6opr3p)

When I wrote this back in 2008, I firmly believed that open source was dangerously veering toward an unsustainable state. After all, enterprise adoption of open-source software was booming as the global economy tanked, but the same companies that were happy to use open source were usually not willing to contribute back.

Soon after, however, the industry changed significantly, paving the way for the biggest boom in open source software development in history. The reasons are two-fold: first, open source licensing strategies became much more sophisticated, and second, a new breed of enterprise arose that gleaned significant benefit from giving away open source software without needing anything in return.

Open Source Licensing Grows Up

The early years of open source were marked by fractious religious wars between advocates of free software and open source software. The free source crowd looked to the GNU General Public License (gnu.org/licenses/gpl.html) as the ideal license to enforce user freedom, because it forced iron-clad guarantees that the code in question would remain open source, while the open source group focused on a broader definition of freedom, preferring the more liberal Apache license (apache.org/licenses/). While the GPL took centre stage during the formative years of the free and open source software movement, governing the development of Linux and other significant projects, over time it has given way to a trend toward Apache-style licensing.

The reason? Developers.

Adherents of GPL-style licensing continue to insist that all software *should* be free and the license must manacle any attempts to extend it with proprietary software. Unlike the Apache license, the GPL embeds the decision as to the code's open source nature into the code itself: if you use the software, you *must* release any derivative works as open source. You have no choice. Apache is very different. Apache adherents believe that software *can* be free and is perhaps best when free. But, these same adherents are not prepared to force other developers to agree with them, and the license does not embed a final decision into the code itself. Downstream users of Apache-licensed software are given wide latitude as to how they use (and license) the software.

As the importance of developers has grown in the software industry, Apache-style licensing has boomed, outpacing GPL-licensed projects to a considerable degree. These trends are illustrated in Figure 1, which is reproduced from Matthew Aslett's (2011; tinyurl.com/7ujq7sj) blog post on this topic.

In many ways, this decline reflects a rejection of the premises underlying free software licensing, with its rigid focus on *software freedom*, in favour of a broader

Is Open Source Sustainable?

Matt Asay

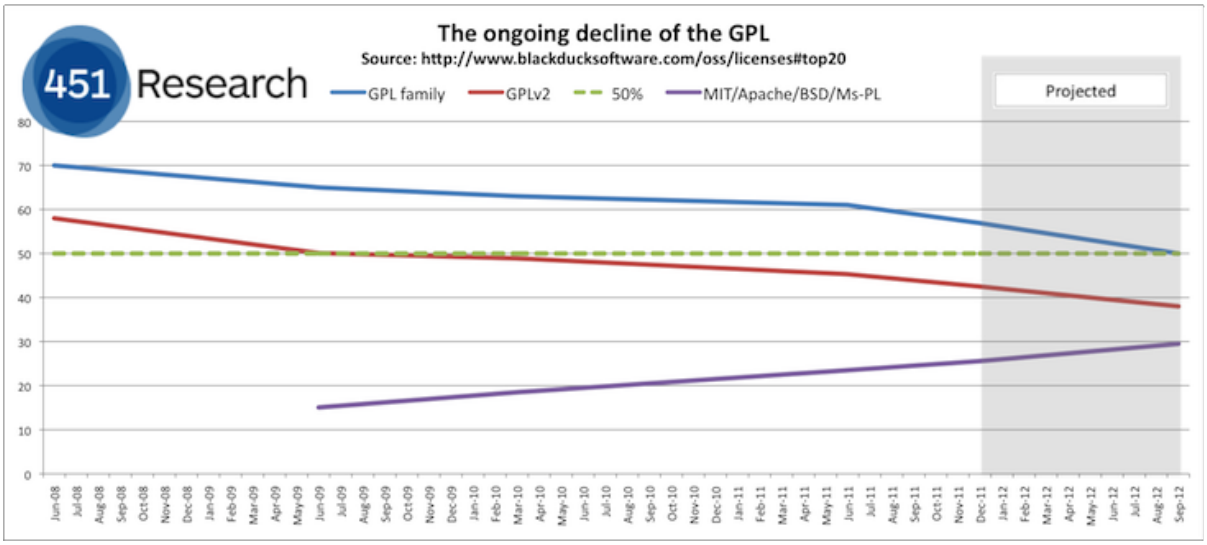

Figure 1. The rise of Apache-style licensing and the decline of the GPL since 2008. Used with permission by 451 Research.

emphasis on *developer freedom*. It is also a rejection of the early open source business model, which used the GPL to essentially build a proprietary software business from free software licensing. That is, with the GPL, an open source vendor could give away its software under a license that many viewed as radioactive, to the extent that it was completely open… and effectively proprietary. Given that the GPL requires any derivative works to also be licensed under the GPL, including third-party software that links to GPL software, depending on how the developer links the software, the GPL puts fear into the heart of legal counsel of would-be users and commercial developers of GPL software. It is free to use, but the possibility of "tainting" one's code is a risk many simply refuse to take. As such, it is open but closed to such companies, that is, impossible for them to accept using without purchasing a proprietary license to use the GPL software.

The seeds of this trend toward permissive licensing were planted in the mid-2000s as legal departments within large enterprises tried to figure out ways to safely embrace open source software. The GPL and its peers nearly always raised red flags, but Apache and its ilk were given a green light. If you were a developer working within Citi Group or Electronic Arts, it was much easier to get a project done with Apache-licensed open source software than GPL-licensed software, because Apache-licensed software makes essentially no demands on users of the software, putting the hearts of legal counsel at ease.

It was not that developers only took their cues from their legal departments. The exigencies of the GPL weighed down development, requiring a degree of license management that was as burdensome to the developer as it was frightening to the attorney. To the mainstream developer without a political axe to grind, Apache offered the path of least resistance to getting work done. This was as true for the solo developer as the corporate developer.

The Web Giants Get Involved

Even as the traditional enterprise grappled with the licensing issues imposed by free and open source software, a new breed of enterprise sidestepped these issues completely. Facebook, Google, Twitter, and other web companies did not distribute software, and so they were able to freely use both Apache-licensed and GPL-licensed software without bothering about contribution requirements. For years, they did just that, scaling out massive infrastructure on open source software, such as Linux and MySQL, that they modified but did not contribute back to.

Not much, anyway.

Facebook changed all this. Facebook's attitude toward open source has always been one of "freely given, freely give," even as Google and Yahoo! kept much of the open source modifications they made to themselves, arguing that few companies besides direct competitors

Is Open Source Sustainable?

Matt Asay

were in a position to use their software effectively. Facebook openly contributed to open source projects such as MySQL and PHP for some time before it started to release its own innovative open source projects such as Cassandra (cassandra.apache.org).

Since Facebook set the tone, Google, Twitter, and others have followed, releasing some of the industry's most promising software, such as Hadoop (hadoop.apache.org), Storm (storm-project.net), and many other projects. Unlike their more traditional cousins in banking, retail, or other industries, these web giants do not really view software as a competitive differentiator, but instead believe that operating this software at scale is what distinguishes them. They also do not have to worry about directly monetizing open source software, so they can release fantastic software with an eye toward developer adoption, not revenue.

This new strategy was a big upgrade over an earlier phase of open source business strategy, which saw venture capitalists funding open source equivalents to BEA Weblogic (tinyurl.com/76a529v) or Siebel's CRM system (tinyurl.com/383xnjh). Although such open source companies did a great deal to move open source forward, proving that it could be useful for a wide array of enterprise-class applications, theirs was an inferior business model compared to what Facebook and its web peers offered. The web giants sold advertising or other services that happened to be powered by open source software running in a remote data centre. They did not have to worry about selling the software, which gave them every incentive to open source their software.

However, these early open source companies have not stood still. Taking their cue from the web companies, open source vendors such as Cloudera (cloudera.com) increasingly contribute heavily to a core open source project and then sell complementary proprietary software or services. This strategy allows them to contribute fully and without conflict to their chosen open source projects, even while making more money. Not surprisingly, since their primary aim is now developer adoption of these core open source projects, and not direct monetization of them, such companies generally turn to Apache-style licensing.

Where Do We Go from Here?

As companies increasingly turn to open source to drive development and not direct revenue, the incentives are mounting for more and better code to be released under permissive open source licenses like the Apache or MIT licenses. This, in turn, will spur more open source development. In many ways, we are entering a golden age for open source, when projects such as MongoDB (mongodb.org), Hadoop (hadoop.apache.org), and Storm (storm-project.net) push the envelope on innovation, rather than following in the footsteps of proprietary software companies.

Even so, while this almost certainly points to years and years of sustainable open source development, it does not yet resolve the continued inefficiency of software development. At least, not enough. As Red Hat CEO Jim Whitehurst has argued:

> *The vast majority of software written today is written in enterprise and not for resale. And the vast majority of that is never actually used. The waste in IT software development is extraordinary.... Ultimately, for open source to provide value to all of our customers worldwide, we need to get our customers not only as users of open source products but truly engaged in open source and taking part in the development community.* (tinyurl.com/bc2dcxy)

So long as enterprises see themselves as islands of productivity rather than communities of developers, I am not sure that this will change. However, what we are seeing is enterprises gravitating toward common pools of development (e.g., Linux and Hadoop). While it is unfortunate that enterprises are not also collaborating on application software such as CRM or ERP systems, perhaps that is the step they will take once the industry more or less standardizes on the same infrastructure.

In the meantime, expect to see accelerating velocity toward permissive licenses as the race to build communities of developers intensifies. This approach is easier and more effective with permissive licenses such as the Apache license. Even in a world where software is run, not distributed, the nagging doubt imposed by the GPL is simply not worth the bother.

In April 2009, Linux founder Linus Torvalds told me, "There is no upside to pushing freeloaders away." Although he used the GPL as the license to govern Linux, his comment was in response to a question about the desirability of tightening the GPL to block companies such as Google and TiVo from using free and open source software without contributing back. To some, this was freeloading. To him, it was simply how open source works, with value being created by contributions of code *but also merely by the act of running one's code.*

Is Open Source Sustainable?

Matt Asay

This mentality has blossomed in recent years. It may have started with developers hoping to foster large communities around their projects, but it has since hit overdrive with the large web companies who have nothing to lose and everything to gain from developers adopting, building on, and even "freeloading on" their software.

This is the future of open source: wide open…and more sustainable than ever before.

About the Author

Matt Asay is Vice President of Corporate Strategy at 10gen, the MongoDB company. Previously he was SVP of Business Development at Nodeable, which was acquired in October 2012. He was formerly SVP of Business Development at HTML5 start-up Strobe (now part of Facebook) and Chief Operating Officer of Ubuntu commercial operation Canonical. With more than a decade spent in open source, Asay served as Alfresco's general manager for the Americas and Vice President of Business Development, and he helped put Novell on its open source track. Asay is an emeritus board member of the Open Source Initiative (OSI). His column, Open...and Shut, appears three times a week on *The Register*. You can follow him on Twitter @mjasay.

Citation: Asay, M. 2013. Q&A. Is Open Source Sustainable? *Technology Innovation Management Review*. January 2013: 46-49.