ThinkWeek

2008

Forges and Foundations

Over the past decade, the expectations of individual developers, and the commercial interests serving them, have changed. Developers expect free software, provided by non-profit software foundations, and hosted in online forges that grant unrestricted access to source code. These expectations, fueled by developer interest to learn new skills and work on complex projects, pose significant challenges to traditional software companies who have built strong revenue streams by distributing binary software. This analysis will deconstruct the 'open source movement' around five key pillars of business strategy, offer a forecast of the next 3 years, and arrive at a specific set of recommendations for Microsoft to accelerate growth in this space.

AUTHOR:

Jamie Cannon

Forges and Foundations

Author: jcannon
Date: 2/8/2008
Feedback Recipients:

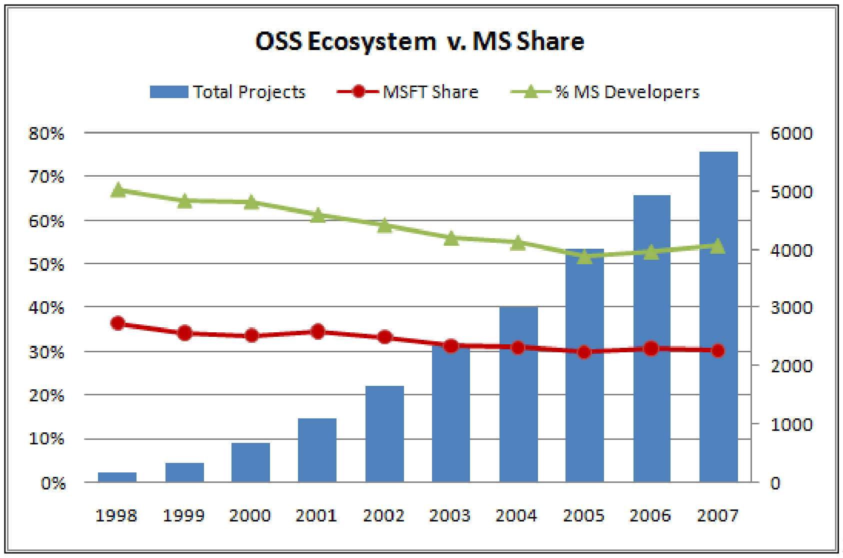
Alias	Title	Description of Role
bryankir	Director	Platform Community Strategy
jonr	Director	Source Access Programs/Unlimited Potential
rebward	Senior Attorney	Source Code Access Legal Counsel
jimmark	Senior Attorney	Open Source Legal Counsel
billhilf	General Manager	Windows Server & Competitive Strategy
tonyhey	Corporate Vice President	Microsoft Research

Introduction

Over the past decade, the expectations of individual developers, and the commercial interests serving them, have changed. Developers expect free software, provided by non-profit software foundations, and hosted in online forges that grant unrestricted access to source code. These expectations, fueled by developer interest to learn new skills and work on complex projects, pose significant challenges to traditional software companies who have built strong revenue streams by distributing binary software.

The press summarily calls these trends the 'open source movement.' Our internal culture has cemented them as competitive threats. Are these accurate characterizations? Are we adequately equipped to capture the developer mindshare currently driving $1.6B in open source revenue[1]?

The graph below begins to answer these questions by highlighting MS share trends in the open source ecosystem over the past decade:



[2]

[1] Worldwide Open Software Business Models, Matt Lawton, IDC. 2007.

Beginning in 1998, OSS projects that ran on Windows and used Microsoft technology represented 36% of all open source projects. This has steadily declined over the past decade, until stabilizing over the past two years at a steady 30%. Further, developers contributing Microsoft technology to this ecosystem represented 67% of total participants in 1998, declined over the same period, and is now resting at 54%.

Only recently have we seen a slight turnaround of these trends; in fact, it is fairly correlated to a similarly timed set of OSS investments from across Microsoft, outlined below. These began in 2005 and continue today as deliberate outreach efforts targeting open source audiences:

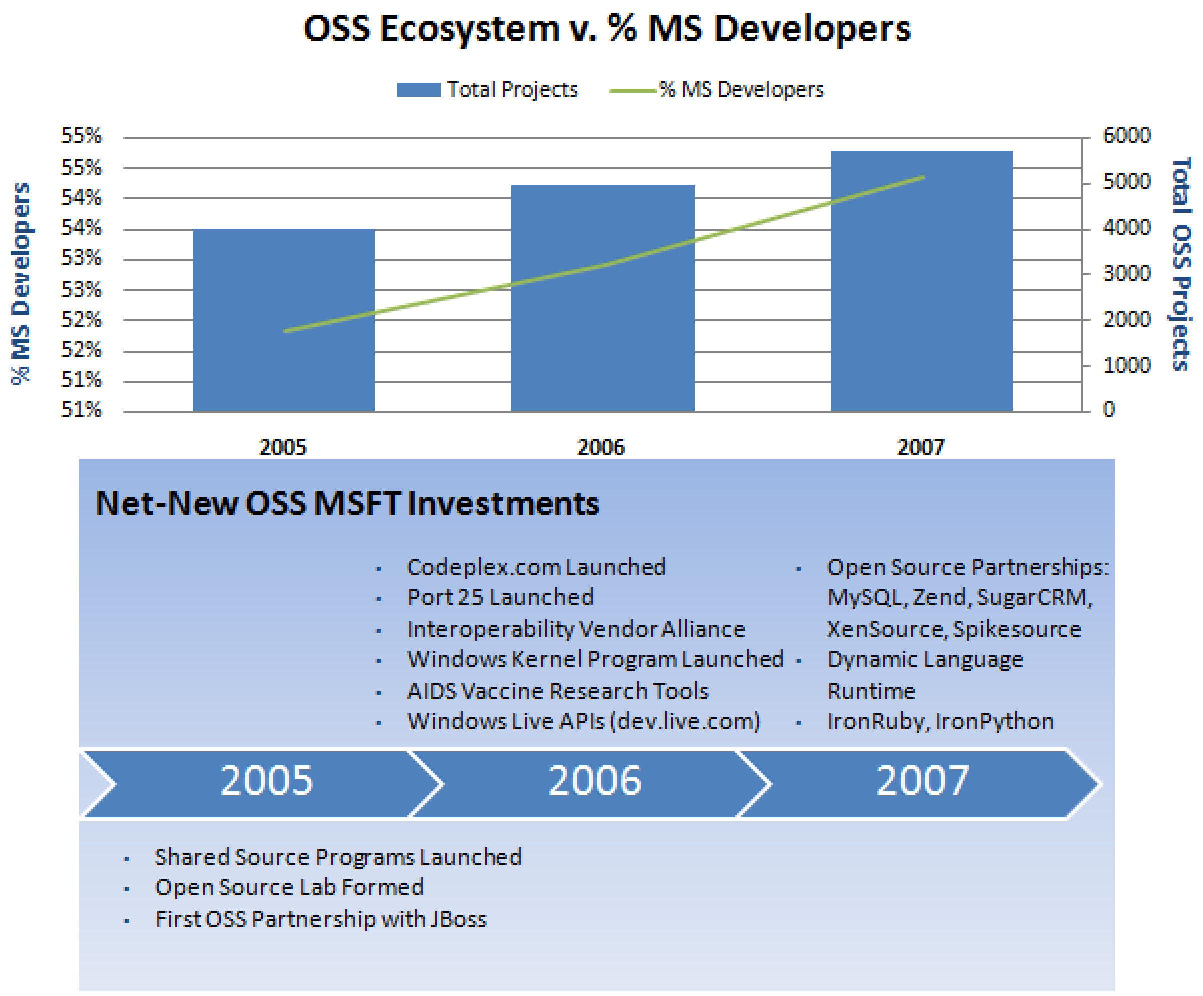


Figure 1: OSS at Microsoft, Timeline v. Developer Share in OSS Projects

Taken collectively, I believe these results demonstrate Microsoft's ability to effectively shift momentum in the 'open source movement'. Coupled with the remaining 70% of OSS project share and projected revenue forecasts, now is the time to accelerate our momentum through expanded

[2] Data provided by Ohloh Corporation. Methodology in Appendix.

use of online forges and non-profit software foundations.

This analysis will deconstruct the 'open source movement' around five key pillars of business strategy, offer a forecast of the next 3 years, and arrive at a specific set of recommendations for Microsoft to accelerate growth in this space.

Intellectual Property Management

Over the past decade, the intersection of the software industry and the subject of intellectual property and patents have become hotly contested issues facing software companies. At one extreme, software freedom extremists advocate enforcement of licensing terms that require all software source code to be freely available and reciprocally shared. At the other end of the spectrum, commercial interests have traditionally been well served by distributing binary software with protected source code. For the purposes of a productive analysis, some core concepts should be defined.

A developer writes code to solve a problem. That solution is an invention. Software inventions are intellectual property because you cannot physically touch them. Typically, the developer, or his employer, owns the invention. With intellectual property, unique inventions are recognized by awarding a patent to the inventor. Thus, a patent represents ownership of the invention.

Inventions can be closely guarded (as a trade secret), or openly shared (as is common in academia or research). This spectrum of sharing is enabled by the concept of licensing – a part of contract law. In a license, a property owner typically grants a set of usage rights associated with their invention to a third-party. Microsoft does this every day through extensive use of End-User License Agreements (EULAs) with our customers.

Ownership of the invention can be transferred through another type of agreement, typically called an assignment, donation or grant of the patent.

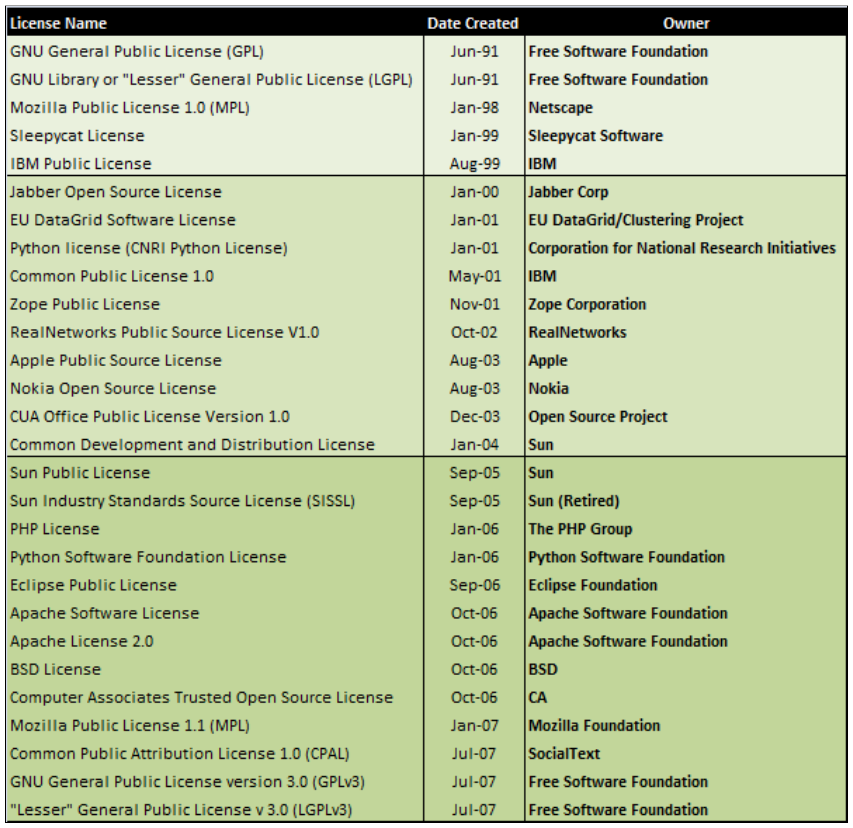
These constructs are not perfect and there is much work underway by Microsoft and others to improve the quality of software patent regulations. However, this represents the current regulatory environment all developers operate under, and in fact – all *property* owners.

What open source has brought to the conversation is a unique copyright structure that binds source code access with licensed software distribution. Many open source licenses go beyond this and also require reciprocal sharing of derivative works. This concept is called copyleft. While enforcement of these concepts can be interpreted as sidestepping the rights of individual developers, it is also an exceedingly effective means of building network effects through software distribution.

Because many of our open source competitors are synonymous with the concepts above, such as Linux, our response to their usage has been competitive in nature. Further, Microsoft product revenue is driven by traditionally antithetical practices. These dynamics have helped fuel public perception that Microsoft's commercial model is not well-suited for open source development – in large part driving the overall downward trend of our ecosystem share over the past decade.

That said, Microsoft has made public investments that begin to move beyond entrenched perceptions. In 2005, we began to embrace new intellectual property structures in efforts to grow the number developers building communities on our platforms. IronRuby and IronPython are examples of this, as is the Windows XML Installer (WiX). All provide source code access and take in community contributions. IronPython, for instance, has been downloaded over 28,000 times; WiX is now a planned component of Visual Studio v.next.

Much of this progress has been enabled by the open source licensing structures and copyleft concepts described above.



License Name	Date Created	Owner
GNU General Public License (GPL)	Jun-91	Free Software Foundation
GNU Library or "Lesser" General Public License (LGPL)	Jun-91	Free Software Foundation
Mozilla Public License 1.0 (MPL)	Jan-98	Netscape
Sleepycat License	Jan-99	Sleepycat Software
IBM Public License	Aug-99	IBM
Jabber Open Source License	Jan-00	Jabber Corp
EU DataGrid Software License	Jan-01	EU DataGrid/Clustering Project
Python license (CNRI Python License)	Jan-01	Corporation for National Research Initiatives
Common Public License 1.0	May-01	IBM
Zope Public License	Nov-01	Zope Corporation
RealNetworks Public Source License V1.0	Oct-02	RealNetworks
Apple Public Source License	Aug-03	Apple
Nokia Open Source License	Aug-03	Nokia
CUA Office Public License Version 1.0	Dec-03	Open Source Project
Common Development and Distribution License	Jan-04	Sun
Sun Public License	Sep-05	Sun
Sun Industry Standards Source License (SISSL)	Sep-05	Sun (Retired)
PHP License	Jan-06	The PHP Group
Python Software Foundation License	Jan-06	Python Software Foundation
Eclipse Public License	Sep-06	Eclipse Foundation
Apache Software License	Oct-06	Apache Software Foundation
Apache License 2.0	Oct-06	Apache Software Foundation
BSD License	Oct-06	BSD
Computer Associates Trusted Open Source License	Oct-06	CA
Mozilla Public License 1.1 (MPL)	Jan-07	Mozilla Foundation
Common Public Attribution License 1.0 (CPAL)	Jul-07	SocialText
GNU General Public License version 3.0 (GPLv3)	Jul-07	Free Software Foundation
"Lesser" General Public License v 3.0 (LGPLv3)	Jul-07	Free Software Foundation

Figure 2: Open Source Licenses, 1998-2008

The Competitive Perspective

The compiled list (above) catalogs some of the open source licenses created and maintained over the past 10 years from a combination of commercial and community competitors. This represents 43% of all open source licenses in the world, and close to the entire body of licenses currently in use by the open source ecosystem. In addition, the rate of OSS license growth is accelerating; 46% of these licenses were created in the last 3 years; 36% in the first half of the decade, and only 18% in the late 90s.

In large part, growth is accelerating due to an increasingly large body of market data that supports successful strategies of growing community and revenue through sharing intellectual property. In doing so, software companies are empowering influential developer communities to coalesce around their platforms.

Red Hat, for instance, is ranked the 11th fastest growing technology company in the US, with revenue of $135M, and is now selling proprietary management technology atop their Linux distribution; Hewlett Packard [3] is shipping 200 open source products with multiple IP contributions to the community, backed by strong fourth quarter net revenue up 15% ($3.7 billion) to 28.3B.

[3] http://opensource.hp.com/

MySQL has employed dual-licensing structures for years, recently hitting the $50M revenue mark with a subsequent acquisition by Sun; Apple openly licenses many components to OS X while protecting core innovations.

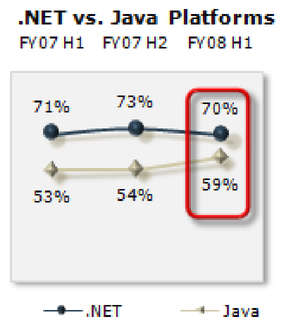
Another example of capitalizing on open intellectual property trends is IBM's use of the Open Innovation Network, a holding company for patents that provide royalty-free licensing to any entity willing to relinquish patent claims against Linux. In 2005, IBM publicly donated 500 patents to the OIN. Headlines were made and to this day, there exists tremendous developer goodwill towards IBM as a result of that patent donation.

However, the donation itself represented only a fraction of IBM's portfolio of over 41,000 U.S. and nearly 30,000 European patents (at the time) – a portfolio which is growing at over 3,100 patents a year. Further, many of the donations are junk patents[4], such as a "Computer system and method for performing multiple tasks" or "a tool for defining a complex system." IBM recently repeated this play by sponsoring the creation of Eco-Patents Commons with an additional donation of 31 environmentally-orientated patents, bolstered by industry support from Sony and Nokia. In both cases, the donations generated overwhelmingly positive press and community reaction with very little actual sacrifice.

Perceived sacrifice is key in driving community goodwill. As a multinational reporting 2007 full-year revenue at 98.8 Billion, IBM understands their position as a leader and understands the outside-in view of a leader who gives back to the communities from which it gains. They use this as a competitive strategy to drive public image and incite developer motivations – regardless of the fact that their donations are oftentimes useless. It doesn't matter if the sacrifice is material, it is the intent of sacrifice that matters.

IBM's Cloudscape database technology is another example of effective copyleft strategy, as is Cloudscape's connection to both Websphere and Rationale. Cloudscape is available as an open source project through the Apache Derby Project. It is also designed to support Websphere Application Server Community Edition. WAS CE is free, and also available under open source terms through the Apache Geronimo[5] project – IBM supports both Derby and Geronimo. Finally, both are well integrated with Eclipse, an open source IDE written primarily in Java and supported by the Eclipse Foundation. Coupled with Linux, this begins to offer complete alternatives to Microsoft developer tools and is intended to do just that. By removing friction to trial, IBM knows they have a better chance at breaking the network effects of Microsoft's developer platform. As noted by IBM in a recent partner webcast, "Of those opportunities that would have never have come about without the existence of WAS CE, 50% resulted in the sale of other WAS products, like WAS Network Deployment." [6] It's all about planting the seeds of proprietary technology wrapped in open source loss leaders.

[4] The full list of donated patents is here.
[5] IBM employs 19 of the 43 commiters working on Geronimo.
[6] Robert Arfman, Director, IBM Business Partner Linux Initaitives



Tactics like this, coupled with support of the Eclipse Foundation, drive market results. In the most recent Developer Tracker Survey, released in November 2007, worldwide professional developers reported overall flat .NET growth at 70% share, with a 5% gain in Java during the same period (right).

Microsoft is moving in the right direction, significantly increasing the capacity of Shared Source to publish code into the community, seeking and receiving OSI approval for two of its own open source licenses, the Microsoft Public License and the Microsoft Reciprocal License, and instantiating a home for shared source code (Codeplex).

We have great opportunity ahead of us– by virtue of our ecosystem size and our unprecedented innovation leadership - to leapfrog IBM's public posturing and community goodwill. To do so, however, we must embrace a more complete intellectual property strategy of ***community*** and ***sacrifice*** to truly have a competitive chance at winning back developer mindshare.

Participation

One of Microsoft's company quests reads: "Changing the Definition of Developer" and in more detail it describes a commitment to taking on big technology challenges that require complex engineering so that developers do not have to focus on 'plumbing.'

This is a noble quest and is very much alive today: our innovations in the .NET Framework, Dynamic Languages, DirectX and throughout Windows have increased the ability for developers to take advantage of well-tested, reliable, supported and documented platform services. And in lieu of spending time in the plumbing, developers are investing more time engineering software that uniquely interests them.

While this increases capacity for Windows to host unique applications (and it does – our ecosystem of 750,000 business partners drives unprecedented application availability, which in turn drives many customers to choose Windows,) it does not address a growing segment of developers that is increasingly characterized by:

- Developers who *want* to play with complex system-level code to learn new skills.
- Student-developers coming out of school with an expectation that open source will help them build programming experience and resume credentials.

There is a growing body of research to support these claims: the EU conducted a thorough analysis [7]of OSS developers wherein most indicated practical motivations to join and continue in the community. "The single most important motivation was "to learn and develop new skills." Nearly 31% noted career and monetary concerns.

[7] Economic impact of FLOSS on innovation and competitiveness of the EU ICT sector. Rishab Aiyer Ghosh. 2006.

In another study conducted by Harvard's Karim Lakhani, increased developer interest in open source participation is explained by the fact that the, "largest and most significant determinant of effort (hours/week) expended on a project (coding) was an individual sense of creativity felt by the developer…More than 60% rated their participation as the most creative experience in their lives."[8] The ability to tinker while learning is a uniquely stimulating experience.

Across all segments of the Lakhani study, an average of 45% were motivated by, "coding that is intellectually stimulating to write." Lakhani goes on to identify these motivations as "enjoyment-based intrinsic motivations." Alternatively, he also identifies, "economic/extrinsic-based motivations," where over 40% are motivated by improving their programming skills (professionally.)

Further, the belief that the majority of OSS developers want code to be open source based on ideological grounds is false. In fact, in a survey of 684 OSS developers, only 11% were found to be motivated to contribute to OSS development because of, "dislike of proprietary software and want to defeat them."

Layered atop this dynamic is a segment of developers that Microsoft cannot ignore – the next generation of ISV developers. Many of these individuals are students coming out of university, looking for apprenticeship on open source projects. In the 2005 analysis of "Understanding Free Software Developers: Findings from the FLOSS Study," Rishab Ghosh notes that nearly 69% of open source developers are under the age of 26.

While Microsoft has programs to land technology into student hands, we do not have robust communities sharing source code and connecting developers around code. Again, competitors, such as Google understand this dynamic as demonstrated by the Summer of Code program – a virtual internship program that gives students around the world a chance to work on Google source code. In 2007, according to Google, the Summer of Code[9]: "…brought together 900 students and nearly 1,500 mentors across 90 countries to contribute to over 130 different open source software projects."

How can Microsoft offer better opportunities for developers to participate with code; how could we expand inclusion to eager students? Today's answer is scattered between Codeplex, the Microsoft Developer Network (MSDN) and a handful of discrete academic evangelism programs.

The possibility of a singular community of code at Microsoft is non-trivial with current online communities at various stages of 'going open.' For example, Microsoft is heavily invested in MSDN as its flagship developer communications platform. Further, it's divisions like Developer and Platform Evangelism where the most progressive work is being done to expand access to our technology and allow for open source development. Scott Guthrie's team in particular has done phenomenal work opening the .NET Platform, in partnership with the Shared Source and Codeplex

[8] Karim Lakhani and Robert Wolf. Why Hackers Do What They Do: Understanding Motivation and Effort in Free/OSS Projects. MIT Press. 2005.
[9] Microsoft does have opportunity to expand the Imagine Cup, but it is currently a very different program.

teams. MSDN itself is increasingly taking steps to towards community-styled features that connect individuals to the people and content they are most interested in. Codeplex is seeing strong growth with over 3,500 open source projects currently hosted.

This is all movement in the right direction. However, a more holistic strategy is needed to unify the open source developer experience at Microsoft; a strategy that removes trial friction through a unified soure code forge and emphasizes the ability of developers to:

- Learn new skills and connect with other developers through source code collaboration.
- Build technology credentials through open source project work.

Technology Access

Today, more intellectual property is being shared than ever before and that availability is exciting developer motivations to share and collaborate together. Taken together, these dynamics have greatly accelerated the deployment of open source software. Projects like Apache run well over 50% of Internet web servers, and usage of the Linux kernel in the enterprise is growing. Google has become a formidable competitor and is doing so through strategic use of open source. From the sound of it, one might assume the entire paid software ecosystem is under seige by open source.

This is a dangerous assumption to make. Open source development is decidedly representative of the long-tail phenomenon. Linux, Apache, MySQL – these are statistical anomalies afforded success by being aggressively opportunistic and decidedly developer-orientated. As Josh Learner and Kean Tirole note in Perspectives on Open Source, "The greatest diffusion of open source projects appear to be in settings where the end-users are sophisticated [like web workloads][10]." So what drivers are enabling these technologies to flourish so completely?

As noted, the majority of community development is occurring in the long-tail. For instance, when one looks at Sourceforge, the world's largest open source forge with over 168,000 projects, only 7% of projects register as active, or roughly 12,000 discrete projects. This distribution is supported by additional data provided by Ohloh, an open source project tracking firm. Again, when activity is measured by a code commit within one year, about 9,000 projects appear to be in active development. The remaining 90% of projects advertised on Sourceforge look more like ad-hoc, deadwood or abandoned efforts.

Within these active projects, most are low-level consumer or developer-orientated applications. There are relatively few deeply complex open source engineering efforts, and those that exist tend to be computer science-orientated, such as compilers, programming frameworks, database engines and debuggers. Linux is the most famous example, the pride of many developers looking to be credited with operating system engineering experience. However, the overall long-tail trend of OSS development looks to remain true into the foreseeable future.

The question remains as to how so many projects are surviving in the wild then. As Robert Glass observes, they are not. Says Glass, "The community movement itself would likely cave in on itself

[10] Economic Perspectives on Open Source. Josh Learner and Jean Tirole. 2005.

over time either over the lack of sustainable economic model, or political splintering."[11] Thus, in efforts billed as "investments to sustain the community," IBM and others harness its power. "IBM has contributed more than $1 billion to the development and promotion of the Linux operating system, and other vendors such as Sun are ramping up OSS efforts and investment." Sun has followed through on that observation, recently open sourcing Java, Solaris and acquiring the open source database vendor, MySQL. Further, through strategic funding, patent donations and non-profit software foundations, many are now living symbiotically off of open source.

Coupled with these capital investments from large technology firms, community developers now have the resources to satiate their motivations, while commercial interests have new channels to seed their products.

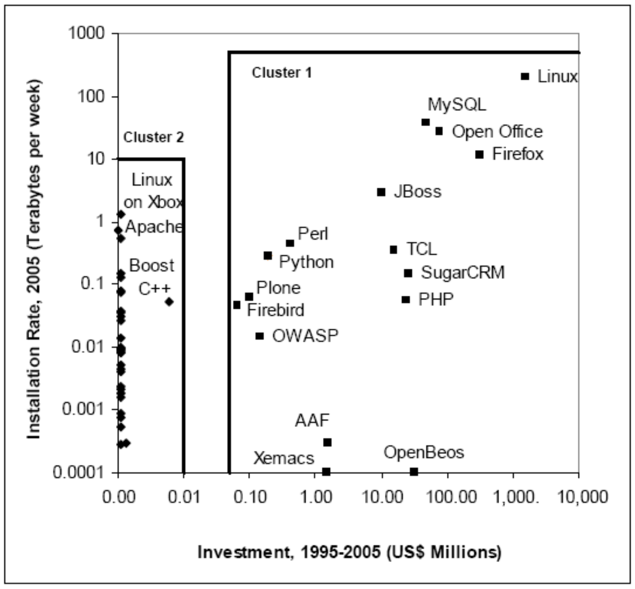


Figure 3: Open Source Investment & Installation Cluster Analysis

Dr. Marco Iansiti calls these commercial companies "Money Clusters," and has done statistical analysis (left) correlating open source investments by commercial firms with their respective download and adoption rates. He observes, "Eighteen projects that received more than 99% of vendor investments" account for the highest installation rate measured in terabyte/downloads per week. Among this list: Linux, MySQL, OpenOffice, Firefox, JBoss, PHP, SugarCRM, Perl and Python.

This begs the question of how these commercial and community organizations, achieve such high rates of download? In large part, is achieved by removing as much friction to trial as possible (registration, authentication, surveying) and publishing projects on globally accessible, unrestricted source code forges. Today, over 1,247 such forges exist, hosting those 12,000 active open source projects.

The experience is seamless. In a few clicks, one can search, browse, download and install almost any open source project I desire. I can do so without registration or demographic profiling, or using a Windows Live ID. I can do so using any browser or computer connected to the Internet. Once in my possession, I can do what I want with it. In many cases, doing so immediately connects me to a community of like-minded developers.

Do we believe we are currently equipped to provide the same simple touch points to our technology? Do we believe our distribution points are adequately platform agnostic to reach new customers?

[11] Standing in Front of the Open Source Steamroller. Robert Glass. 2005.

Legal Risk

The ability to instantiate, participate and contribute to open source is not easy for commercial software companies to do. Corporate entities are formal structures, designed to interact best with other formal structures. Open source projects are typically unincorporated, loosely-coupled volunteer efforts. One Fortune 100 executive interested in a deal with Apache captured this tension best by asking, "How do I make a deal with a web page?"

The answer to this question has given rise to a new kind of commercial entity that now hosts many open source projects – and is increasingly the recipient of commercial donations. This new entity, the public non-profit software foundation, has come into being over the past decade. For instance, Apache answered that Fortune 100 Executive by forming the Apache Software Foundation (ASF) such that it could organize and contract with other property owners. IBM supported the formation of Eclipse; multiple vendors support the Linux Foundation and Free Software Foundation.

So why – beyond developer participation and strategic IP management, should Microsoft look at increasing participation through foundations and forges?

Reducing risk exposure is one reason. As mentioned earlier, the GPL is a popular license in the open source community. It's a fine license and v2 is very effective at what it sets out to do. However, v3 has a more ambitious agenda and is more problematic for companies like Microsoft. In effect, GPL v3 not only requires communal and reciprocal sharing, but now includes language that could expose any "conveyor" of GPLv3 software to provisions which automatically grants patents of licensed inventions as well.

Microsoft risk assessments currently focus on current GPLv3 usage and expected adoption outlook, best reflected by the Open Source Software Development Survey, published by Evans Data Corp in September 2007. Evans reports, "Only 6 percent of developers working on Open Source software have adopted GPLv3." Further, two-thirds say they will not be adopting GPLv3 anytime in the next year, and 43 percent say they will never implement the new license.

The chart (right) illustrates actual GPLv3 growth in the open source project space over the past six months, with a total of 1,774 currently licensed projects on GPLv3, up 17% from 2007 Q4. It is worth noting that the GPL v3 was made available during mid-year 2007.

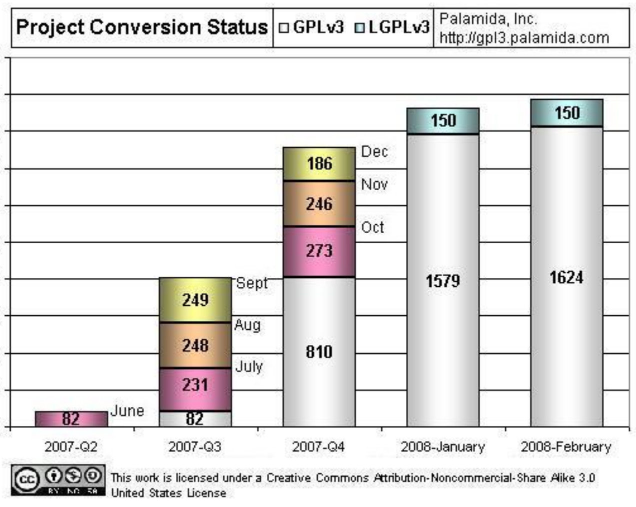


Figure 4: Adoption of GPLv3 Among GPL Projects

With this rate of adoption, over time, GPLv3 will become a common license in the technology landscape. A method is required to assure protection of Microsoft from GPLv3 code. It is my belief that much like IBM, we can capitalize on these trends – and decrease our level of risk – by spinning out a software foundation, initially sponsored by Microsoft, and supported by our ecosystem of partners.

This would simultaneously allow us to increase our participation in the open source community, better serve developer motivations, and do so in a way that better protects (and increases opportunity of) our inventions.

A software foundation can exist in a number of ways: as a non-profit legal entity wholly owned by a for-profit parent organization. The Mozilla Foundation is an example of this, owned by the Mozilla Corporation. This particular model allows the foundation to enter into contracts and execute business transactions, but it cannot accept the common idea of 'individual donations.' This is how, for instance, Google secured search placement through Firefox with a $50M payment to Mozilla. A software foundation can also be self-incorporated and exist independently. Eclipse and the Apache Software Foundation (ASF) are examples of this.

Local Software Economy & Foundations

Foundations – driven by a combination of developer motivations and corporate legal interest – are also increasingly finding a place as software brokers to governments and local economic organizations. While not as common in the US, many governments around the world are embracing open source as a political platform to encourage local IT economy, reduce dependence on foreign multinationals and promote greater autonomy for regional governments[12].

An interesting dynamic begins to form. As noted above, the active open source community could not sustain itself with commercial investment. Communities are increasingly acting through foundations. Those foundations are interacting with governments around the world. And in many cases, commercial interests are now influencing those governments' through the foundations they support. IBM is effective at using the Eclipse Foundation in this way; Apache is driving their own successes with continued sponsorship by Yahoo, Google, HP, Covalent and many others.

In the figure below, a chain of influence is described that outlines the flow of power from commercial sponsors to foundations, and how they act as middlemen between open source communities and local IT growth initiatives.

[12] Open Source in Latin America: The Private Sector Impact on Public Policy. The Yankee Group. 2007.

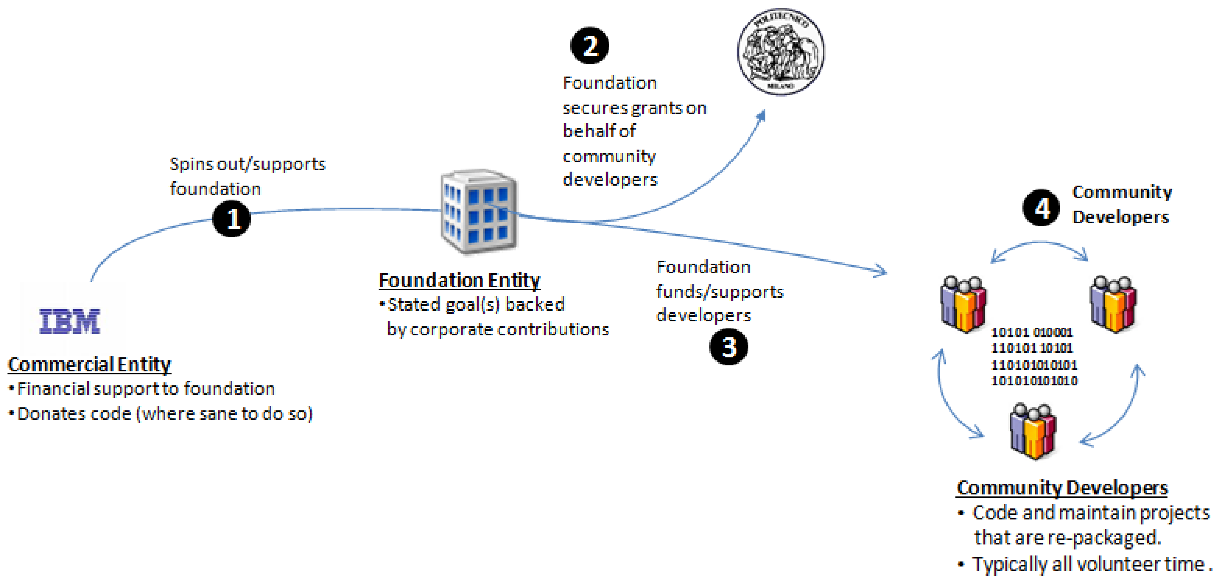

2
Foundation secures grants on behalf of community developers
Spins out/supports foundation
1
Community Developers
4
Foundation Entity
•Stated goal(s) backed by corporate contributions
Foundation funds/supports developers
3
IBM
Commercial Entity
•Financial support to foundation
•Donates code (where sane to do so)
10101 010001
110101 10101
110101010101
101010101010
Community Developers
• Code and maintain projects that are re-packaged.
• Typically all volunteer time.

Figure 5: Chain of Influence, How Software Foundations Drive Commercial Interests

This is an innovative market structure; governments feel comforted that they are meeting the needs of their citizens, companies like IBM are using foundations to work with the community and drive sales; communities are serving developer motivations. This is a significantly creative model we haven't yet harnessed in our own efforts – and stand to gain from in our Unlimited Potential and local software economy initiatives.

Platform Economics

Microsoft is a platform company dedicated to creating, building, and supporting broad ecosystems of developers and partners who deliver compelling experiences and solutions to customers. Success of this business model depends on producing software that is generative: the technology's overall capacity to *enable third parties* to generate new and valuable uses to meet the needs of large and varied audiences[13].

As elaborated in this analysis, the IT ecosystem is shifting away from pure command and control structures. Instead, centralized power is being replaced by the greater capacity of innovation and creation associated with "...the power of user innovation, independent of any manufacturing firm." That, in fact, markets characterized by user innovation, "have a great advantage over manufacturer-centered innovation development systems that have been the mainstay of commerce for hundreds of years.[14]"

[13] Invisible Engines: How Software Platforms Drive Innovation and Transform Industries. Harvard Business School, Working Knowledge, 2006.

[14] Open Source Software Projects as User Innovation Networks. Eric von Hippel. 2005.

What's relevant here is that in all of these trends, open source substantially increases the ability of a platform to grow by virtue of the fact that it's not 'directed' growth. In other words, loose restrictions on the use of source code allow a developer to invest in a platform while simultaneously fulfilling any coding curiosity they may have. It is a method for a platform to "expect the unexpected". This is in sharp contrast to traditional platform economics which, to-date, focus on manufacturer-directed control of third-party opportunity.

Expanding platform opportunity is in Microsoft's heritage, and expanded participation through developer forges and foundations would further our ability to incentivize, share and diffuse innovations across Microsoft platforms.

Conclusion

Each of these trends presents unique challenges and great opportunity for Microsoft. What is currently a $1.8B market will grow over the next four years to an estimated $5.8B, with a compound annual growth rate of 26.2% through 2011,[15] according to IDC.

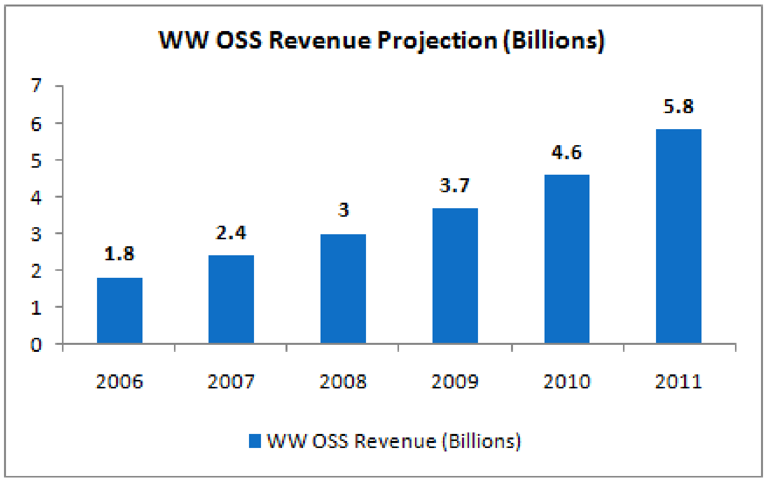


Figure 6: IDC Projection. May 2007.

I would further this with a few additional predictions. As the open source ecosystem grows, momentum of open source on Windows will grow proportionally. This will happen organically over the next three years as a result of continued student, professional developer and government interest in source code access. The larger of the Windows-favorable OSS projects will seek partnerships with Microsoft. This will present a unique opportunity to seed our platform into the open source ecosystem and capture both the revenue and developer mindshare discussed above.

However, should we lack an accessible touch point to engage with, such as a non-profit software foundation dedicated to growing open source on Windows, they will quickly turn to other

[15] Worldwide Open Source Business Models. IDC. May 2007.

formidable sources of commercial support, such as IBM and Google. Further, GPLv3 adoption will also grow, forcing Microsoft into a more awkward position that we need to figure out long-term.

In conclusion, I'm proposing a set of possible starter ideas to begin scoping how Microsoft might harness the power of foundations and forges:

1. Recruit, support and market open source applications on Windows that target K-12 scenarios – helping to bolster our end-to-end education experience on Windows. The theme of education could go much further, working with community developers and organizations to developer interoperability code for various new classroom scenarios, such as interop between Windows and OLPC.

2. Partner with Microsoft Research on an open source foundation; examine areas of incubation, such as a proposed 'eResearch Platform' based on our product platforms with open source or free plug-ins that enhance the ease that scientists can use our tools to support their research. This goes all the way from data acquisition and data mining to scientific publication and archiving.

3. Use a foundation to catalyze the commercial open source community around Microsoft platforms. This would include traditional partners, like HP and Intel – who are both investing in open source – to a new set of technology partners. Donate our own low-value patents to get in the game.

In any one of these cases, I believe Microsoft could fuel new growth in developer mindshare and commercial market share.

Appendix

Ohloh Data Analysis (OSS Projects):

We define a "Microsoft-specific project" to mean "a project that has made affordances to target any of the major Microsoft platforms." Since the data is tracking almost 10,000 projects, the discovery process was automated. We multivariate heuristic was devised that took into account several dimensions, including:

- Project descriptions: we included projects that contained Microsoft-specific keywords, such as "Windows XP", "dotnet", "Microsoft, "XBox", etc...
- Project programming language: we included projects that contained significant amounts of Visual Basic, C# or Batch scripts.
- Source code files and directories: we included projects whose source code trees included some tell-tale files and directories, such as "win32", or with very Microsoftspecific file extensions, "such as .chm".
- Source code analysis: we examined every line of C and C++ based source code in every project, across the entire history of each project, to look for some very Microsoftspecific code signatures.
- We looked for Windows headers (such as "stdafx.h"), wellknown win32 apis (such as "CreateWindowEx") and a few other indicators - (preprocessor directives like #ifdef WINNT).

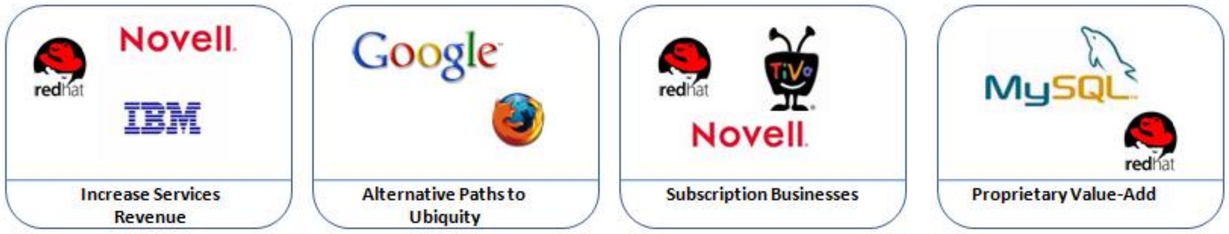
The results were tallied and sampled by hand. A formula was arrived at on to make a final assessment of "Microsoft-Specific" or "Non-Microsoft" projects. In many cases we were able to determine exactly when a project began supporting Windows. In these cases we treated the project as "Non-Microsoft-Specific" until it switched to supporting Windows, at which point we accounted for it as a "Windows-Specific" project.

What is a foundation?

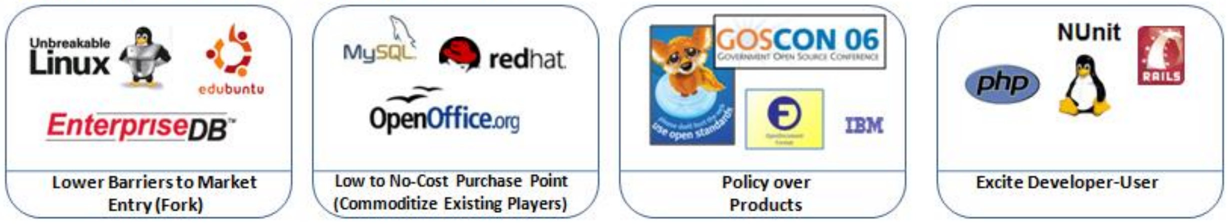
A foundation is an entity that is established as a nonprofit corporation or a charitable trust, with a principal purpose of making grants to unrelated organizations or institutions or to individuals for scientific, educational, cultural, religious, or other charitable purposes. This broad definition encompasses two foundation types: private foundations and public foundations. The most common distinguishing characteristic of a private foundation is that most of its funds come from one source, whether an individual, a family, or a corporation. A public foundation, in contrast, normally receives its assets from multiple sources, which may include private foundations, individuals, government agencies, and fees for service. Moreover, a public foundation must continue to seek money from diverse sources in order to retain its public status.

Open Source Approaches to Business Growth

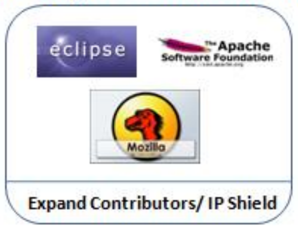
Grow Revenue



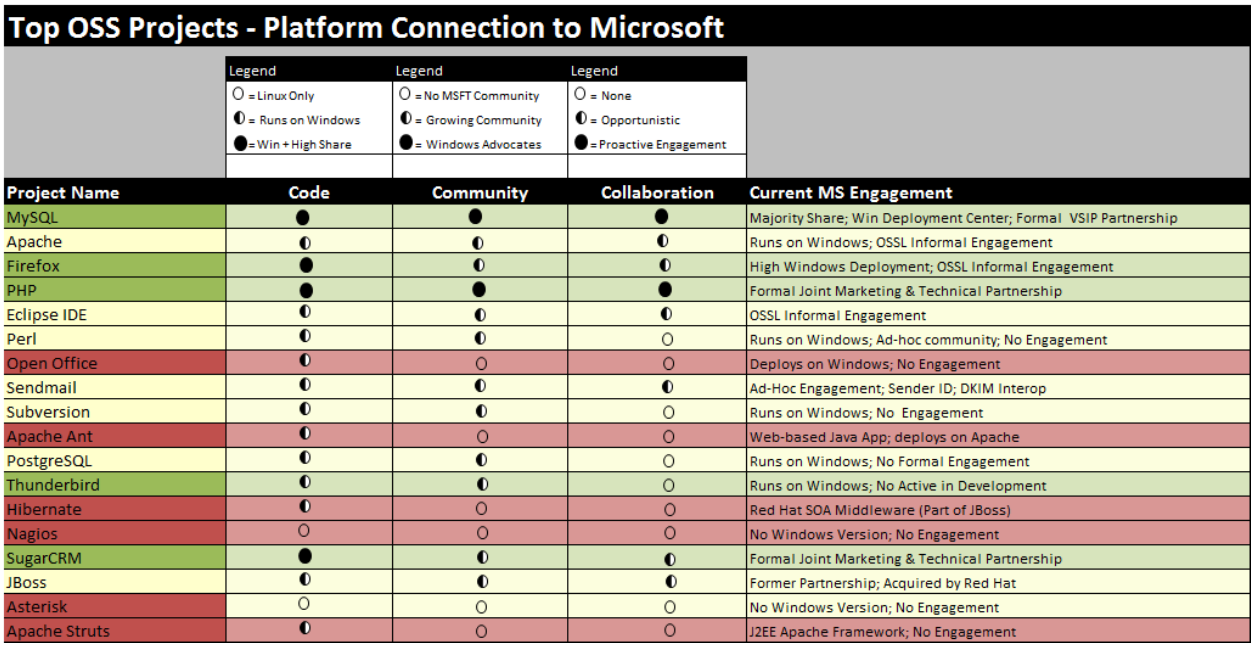
Competing to Win: Weaponizing Open Source



Driving the Business: New Methods of Collaboration


eclipse
The Apache Software Foundation
Mozilla
Expand Contributors/ IP Shield

Current State of OSS Projects on Windows:

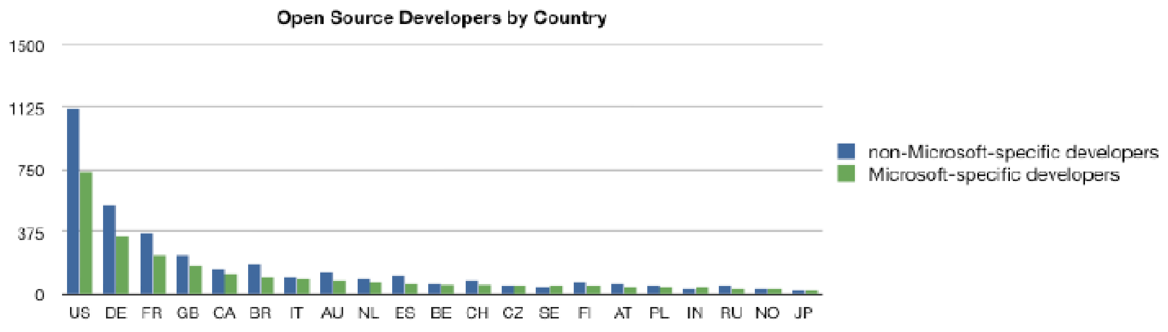


Top OSS Projects - Platform Connection to Microsoft

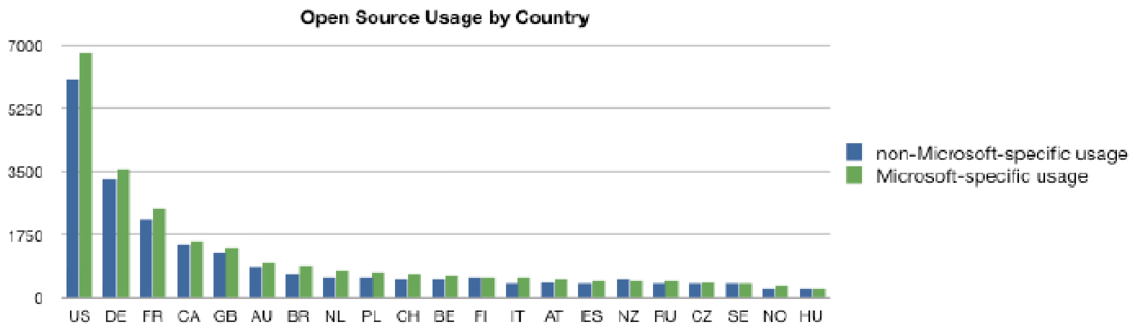
Legend (Code)	Legend (Community)	Legend (Collaboration)
○ = Linux Only	○ = No MSFT Community	○ = None
◐ = Runs on Windows	◐ = Growing Community	◐ = Opportunistic
● = Win + High Share	● = Windows Advocates	● = Proactive Engagement

Project Name	Code	Community	Collaboration	Current MS Engagement
MySQL	●	●	●	Majority Share; Win Deployment Center; Formal VSIP Partnership
Apache	◐	◐	◐	Runs on Windows; OSSL Informal Engagement
Firefox	●	◐	◐	High Windows Deployment; OSSL Informal Engagement
PHP	●	●	●	Formal Joint Marketing & Technical Partnership
Eclipse IDE	◐	◐	◐	OSSL Informal Engagement
Perl	◐	◐	○	Runs on Windows; Ad-hoc community; No Engagement
Open Office	◐	○	○	Deploys on Windows; No Engagement
Sendmail	◐	◐	◐	Ad-Hoc Engagement; Sender ID; DKIM Interop
Subversion	◐	◐	○	Runs on Windows; No Engagement
Apache Ant	◐	○	○	Web-based Java App; deploys on Apache
PostgreSQL	◐	◐	○	Runs on Windows; No Formal Engagement
Thunderbird	◐	◐	○	Runs on Windows; No Active in Development
Hibernate	◐	○	○	Red Hat SOA Middleware (Part of JBoss)
Nagios	○	○	○	No Windows Version; No Engagement
SugarCRM	●	◐	◐	Formal Joint Marketing & Technical Partnership
JBoss	◐	◐	◐	Former Partnership; Acquired by Red Hat
Asterisk	○	○	○	No Windows Version; No Engagement
Apache Struts	◐	○	○	J2EE Apache Framework; No Engagement

Where are OSS developers?



This chart shows open source developers grouped by Microsoft-specific development or not. We categorize a developer as being Microsoft-specific if he or she has contributed to at least one Microsoft-specific open source project. We restricted the chart to show the 20 top countries (the supplemental attachments contain all the data).


Open Source Usage by Country
US DE FR CA GB AU BR NL PL CH BE FI IT AT ES NZ RU CZ SE NO HU
non-Microsoft-specific usage
Microsoft-specific usage