O'REILLY®

Open by Design

The Transformation of the Cloud through Open Source and Open Governance

Philip Estes & Doug Davis

O'REILLY®

Additional Resources

4 Easy Ways to Learn More and Stay Current

Programming Newsletter
Get programming related news and content delivered weekly to your inbox.
oreilly.com/programming/newsletter

Free Webcast Series
Learn about popular programming topics from experts live, online.
webcasts.oreilly.com

O'Reilly Radar
Read more insight and analysis about emerging technologies.
radar.oreilly.com

Conferences
Immerse yourself in learning at an upcoming O'Reilly conference.
conferences.oreilly.com

Open by Design

The Transformation of the Cloud through Open Source and Open Governance

Philip Estes and Doug Davis

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

Open by Design
by Philip Estes and Doug Davis

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://safaribooksonline.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or corporate@oreilly.com.

Editor: Nan Barber
Production Editor: Dan Fauxsmith
Proofreader: Rachel Head
Interior Designer: David Futato
Cover Designer: Ellie Volckhausen
Illustrator: Rebecca Demarest

September 2015: First Edition

Revision History for the First Edition
2015-09-28: First Release
2015-12-07: Second Release

978-1-491-94109-6

[LSI]

Table of Contents

Introduction

If "software is eating the world," then maybe we can say that open source software is devouring it. While open source software is no new kid on the block (look at the rich history of the heavyweights in the room—Linux, for starters), current statistics around community participation, lines of code submitted, corporate involvement, and revenue impact are increasing at amazing rates. At LinuxCon North America 2015 in August, the Linux Foundation announced that over 64.5 million lines of open source code have been contributed into its own umbrella of projects, *not* including Linux itself! These contributions came from thousands of unique contributors, from students to corporate-employed software engineers, to the tune of a rough valuation of US$5.1 billion dollars of software components.

While this is interesting in and of itself, what is possibly more interesting is that open source is not just about lines of code hosted in public online repositories with reasonable open source licenses. Today's open source, managed by open governance and collaborative foundations, is fueling a developer revolution across broad, worldwide communities to solve the next set of computing challenges around the cloud: from infrastructure services to platform and application packaging, to delivery and operational challenges in web-scale production applications.

This open source revolution is changing the landscape of how companies think about developing software, and specifically cloud solutions for their customer base. What we find is that this new era of openness is itself breeding open thinking and collaboration at a massive new scale among experienced developers who formerly were applying their expertise to similar, or even the same, challenges

but within the proprietary confines of their own enterprises. Instead, we now are increasingly seeing openness as an explicit design point in software generally, and cloud computing specifically, for many enterprise organizations that would traditionally have "rolled their own." We are calling this new era the time to be *Open by Design*.

CHAPTER 1

Open Source: A Brief History

What Is Open Source?

To have a reasonable discussion on the topic of open source, we first need to agree on what we mean by the term. After we establish a baseline definition, we'll review a brief history of how and why it exists, and follow its maturation into a viable and valuable component within the development processes of many industries and software domains.

First, while it is valuable for everyone to read and understand the Open Source Initiative's 10-point *open source definition*, clearly one of the most important truths about open source is that access to source code is a necessary but not sufficient component in defining whether any given software is truly open source. As the OSI's definition clarifies, access to source code is a stepping stone that should be followed up with free redistribution—both legally and practically—as well as the removal of roadblocks (discrimination) against disparate (and possibly unpredicted) usage as well as disparate groups of people, both consumers and developers. The best and most valuable open source projects have low friction in all these areas—code access, code sharing, and freedom of use and distribution—allowing ease of use and ease of modification by any and all parties.

It is worth highlighting a key point of the OSI's definition. While there are many open source projects available, simply putting the source code on the Internet is not sufficient. In particular, there are many open source projects that have licenses that make it virtually

impossible for corporate interests to participate in them. This limits the number of developers available to help, and, therefore, the projects' chances for long-term growth and success. For example, a project that requires all derivations of the source code to also be open sourced would be forcing commercial offerings to give their value-add (possibly proprietary) logic away for free. For some, this would be a nonstarter. The most successful open source projects realize the variety of reasons why people might participate in the projects and encourage adoption of their technologies without such strong restrictions.

Beyond having access and rights to source code, truly valuable open source projects are much more than codebases. Valuable open source projects include broad, collaborative communities working together toward a single purpose. A single developer, or even a single company's open source project, may be useful to some degree, but true value comes when a disparate group of interested parties invest themselves in improving the codebase. These additional hands are able to invest time and resources to make the software better tested, better documented, more resilient to errors, and with increased functionality to meet the user's needs and requirements. The original author may have intended all those qualities, but truly the power of open source is for a collective of interested parties to provide their time and expertise to accelerate this maturation at a speed and rate practically unavailable to the original author.

Popularization and Commercialization

While we can definitively say that the modern GNU/Linux and Free Software Foundation–fueled era of open source has its roots in a countercultural shift away from corporate interests, patent portfolios, and legacy closed source and proprietary systems, it would be of interest to look at open source history just prior to that point on the timeline of computing history.

In the 1950s and '60s, many of the early computing systems from IBM, DEC, and others were developed in concert with academia, research institutes, and in some cases the government. This led to initial software operating systems and other key software components being assumed to be shared resources among the user and developer bases—which at this point in computing history tended to be one and the same. Early computer system providers would

deliver their hardware with the entire source code to the software for the systems, including the tools required to modify and build the software. For the IBM 701 mainframe system, this particular sharing of source code led to the SHARE user groups and conferences that continued for several decades. SHARE was a vibrant community of systems programmers and users who shared stories about their issues and problems, and then shared code and additions/changes to solve each other's problems.

While the availability of ubiquitous high-bandwidth networks and ease of worldwide communication were still decades away, these beginnings were the roots of the modern open source movement: a collaborative community sharing solutions, source code, and expertise with others without expectation of monetary renumeration, patent rights, or licensing revenue.

Fast-forwarding to the modern era, the introduction of the GNU project and accompanying free software ideas from Richard Stallman in the 1980s, quickly followed by Linus Torvalds and the Linux operating system in 1991, were milestones that, combined with the increasing ease of network connectivity around the globe and mass communication via access to email, early primitive websites, and code repositories on FTP servers, led to a huge influx of new participants in the open source movement. Linux and various GNU project components provided a free base layer for open source activities. All the tools necessary for participating in open source—compilers, editors, network clients, and additional scripting languages and utilities—were embedded in a single freely accessible operating system environment, thereby significantly lowering the bar for entry and involvement by any party with access to a basic personal computer.

It was soon after this influx of new participants in the mid-1990s that for-profit companies were born out of this grassroots open source movement, including big names like Red Hat, SuSE, VA Linux, Netscape (soon to be Mozilla), and MySQL AB. Not only were new companies formed, but many large enterprises soon saw the value of open source development models and began participating in open source communities, with salaried employees directed toward full-time "upstream" open source work. IBM was an early adopter of this strategy: in 1998 it created the IBM Linux Technology Center, hiring Linux kernel experts and repurposing internal employees to work on the Linux kernel and other upstream open

source projects. The goal was to enable Linux across all of IBM's hardware platforms and enable Linux versions of all of its key middleware products. IBM created Linux versions of its popular enterprise software suites like DB2 and WebSphere, and even traditional mainframe-oriented software like CICS and MQSeries. Many other large enterprises followed suit: Oracle, SAP, HP, Intel, and other companies began working directly on Linux, or enabled many of their hardware or software offerings to run on the Linux operating system. No longer was open source just for the "unwashed hippies" (as they had sometimes been ridiculed) of the free software movement; it had now entered the well-heeled boardrooms of multibillion-dollar corporations.

From those early days of corporate involvement in open source, initial uneasiness around using open source software intermixed with proprietary software and solutions has waned considerably. Today it would be hard to find any software solution, from mobile devices to embedded control systems to enterprise data center solutions, that *doesn't* include some form of open source software. This popularization and commercialization of open source continues apace today, and it has definitely marked a significant place in cloud computing, with the Linux OS as the enabler of web-scale compute resources, followed by many significant open source projects providing the scaffolding around this—from hypervisors to infrastructure management, deployment, and application layer frameworks. The lion's share of these projects are open source both in name and in the open nature of the communities themselves. In some cases, open governance communities via foundations have been created around them as well. But before we turn our attention there, we'll look at the history of industry disruption birthed through open source.

Disruption

Whether they understand it or not, most consumers today are also consumers of open source software. Even consumers who have little technological awareness are reaping the benefits of open source—many unwittingly. A significant share of these end-user benefits come via consumer-oriented devices, from GPS units to wireless home routers to streaming devices like Roku and Chromecast. Android, an open source project as well, is used daily by more than one billion people via smartphones and tablets worldwide. Even on personal computers with commercial operating systems, the use of

open source software like Firefox and Google Chrome continues to grow. Stepping back a layer from the personal user to the realm of the hosting provider, the Apache web server continues to be far and away the top web server across all hosted sites, with another open source project, Nginx, quickly gaining market share. In the context of the Web, we should also mention the huge popularity and growth of the open source WordPress content management platform, through which millions of blog posts are written and delivered daily —many by people who have no knowledge that the underlying platform they're using is open source all the way down to the hardware drivers. Given this basic truth that open source software exists in some form at nearly all layers of software and hardware ecosystems, let's take a brief look at the disruptive force of open source across several key areas over the last 15 years.

Server Operating Systems

Prior to the arrival of Linux, Windows and a long list of commercial Unix variants had the lion's share of the server operating system market. Even in the early days of Linux, the expectation was that enterprise customers would not switch to the fledgling open source operating system, even if it was "free." Of course, as the Linux ecosystem grew and companies formed to offer enterprise-class, supported Linux distributions, the market share picture began to change rapidly. By the end of 2007, IDC reported that Linux had finally broken the US$2 billion barrier in a single quarter and had grown to represent 12.7% of all server revenue. By 2010 the share percent for Linux had grown to 17%, but the breakout moment arrived in 1Q 2012, when the IDC reported that Linux had grabbed 20.7% of worldwide server revenue compared to 18.3% for Unix:

> At the Linux Foundation's annual conference in August, IBM VP Brad McCredie told the crowd something that was probably unthinkable when Linus Torvalds created his new operating system kernel two decades ago.
>
> "The Linux market now is bigger than the Unix market," he said.[1]

Turning our attention to supercomputing for a moment, we see an even more significant shift away from traditional Unix to Linux. In Figure 1-1, note that between 2000 and 2010 the Linux share of the

1 Jon Brodkin, "Linux is king *nix of the data center—but Unix may live on forever," Ars Technica, October 22, 2013.

TOP500 supercomputers operating system market went from around 5% to nearly 90%! Obviously, one of the great strengths of an open source operating system is the ability for researchers and hardware designers to quickly innovate on hardware acceleration features, custom-tuned device drivers, and enhanced kernel technology to rapidly prototype, benchmark, and improve high-performance computing workloads. Needless to say, IBM also invested significantly in Linux, bringing Linux support to its POWER and z Systems mainframe platforms and providing Linux in concert with traditional IBM enterprise hardware strengths in a single package for its enterprise customers.

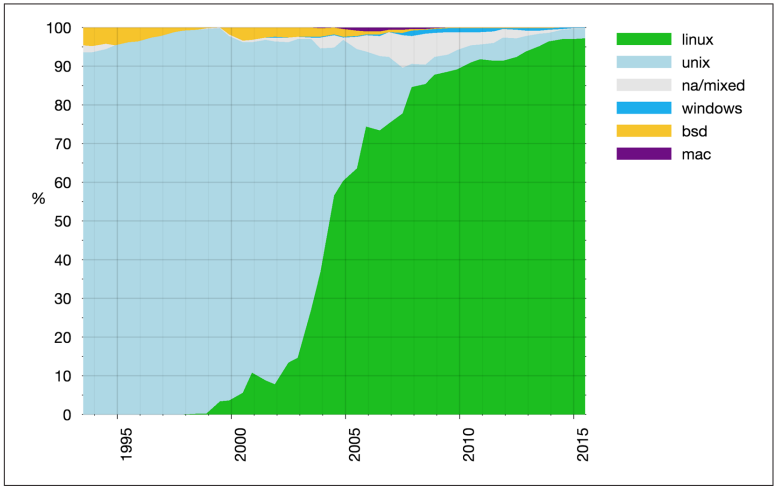


Figure 1-1. Operating systems used on TOP500 supercomputers (source: Wikimedia Commons)

As recently as the latest 2014 report, IDC continues to report year-over-year increases in Linux revenue and server shipments. Looking at worldwide server shipments in 2014, Linux approached 40% share—a 16.4% YoY growth rate—and was bested only by Microsoft Windows, at 59% share on a 4% decline. Interestingly, when looking only at US server shipments in 2014, Linux increased nearly to parity with Windows server shipments, at 48.7% and 50.3%, respectively.[2]

2 Data from "Worldwide and U.S. Server 2014 Vendor Shares," published by IDC on June 5, 2015.

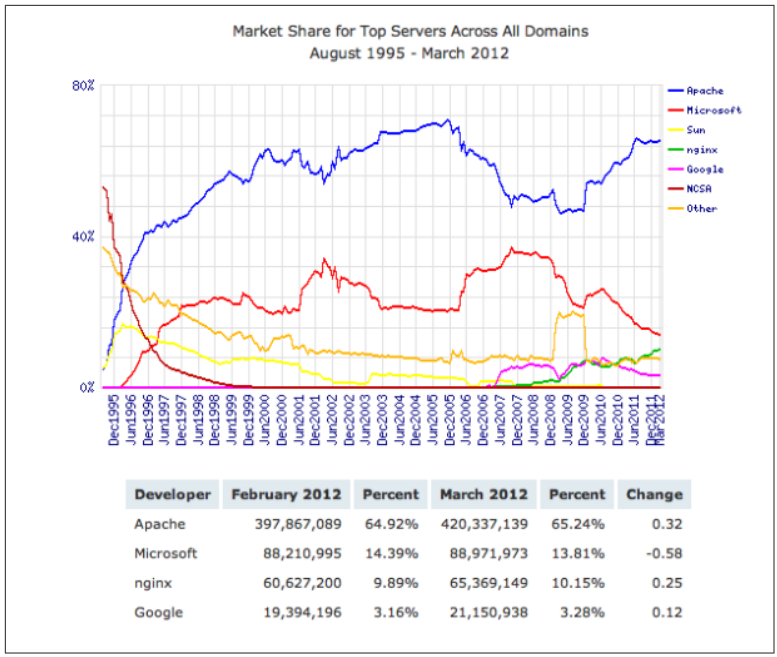
While we can clearly see the disruptive nature of Linux in the server operating system market, it also opened the way for myriad other open source market entrants who quickly followed on the heels of Linux's success. We'll look at a few here, starting with one of the most venerable and long-standing open source software projects broadly used worldwide.

Web Serving

In the early days of the Web, there were few choices for web server software, so the public domain NCSA software developed by Rob McCool was the de facto standard. In the mid-1990s, Microsoft began offering its Internet Information Services (IIS) web server with Windows NT 3.51, and at about the same time, the Apache open source web server project was born. Apache was based on the underpinnings of the NCSA server, which at that point was no longer being maintained. More than having publicly available source code, which was true of the NCSA server, the Apache project was intent on having coordinated development across a set of interested parties, and soon an initial eight core contributors formed the original Apache Group, with more to follow soon after.

In the years ahead, the Apache web server developed into a feature-rich and extensible architecture that was ported and ran across myriad CPU architectures and operating systems. By 1999, the Apache Software Foundation had been formed, formalizing the early community of developers with financial backing, governance, and administrative/legal help. This foundation would soon serve a vast array of open source projects encompassing much more than a simple web server.

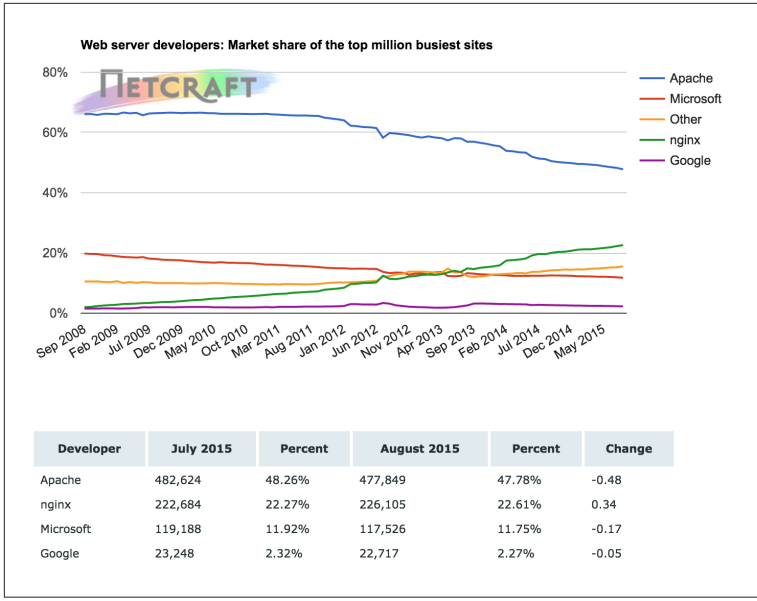
To this day, Apache is far and away the most popular web server platform for hosted Internet sites. Figure 1-2 shows a graph of Apache's dominance in this space, which has continued over two decades.



Developer	February 2012	Percent	March 2012	Percent	Change
Apache	397,867,089	64.92%	420,337,139	65.24%	0.32
Microsoft	88,210,995	14.39%	88,971,973	13.81%	-0.58
nginx	60,627,200	9.89%	65,369,149	10.15%	0.25
Google	19,394,196	3.16%	21,150,938	3.28%	0.12

Figure 1-2. Web server market share, all domains, 1995–2012 (source: Netcraft)

As a postscript to this section, we will show one more graph of web server statistics from the past few years. What Figure 1-3 depicts is disruption again, as yet another open source web server project, *nginx*, is now taking significant market share away from its fellow open source titan, Apache.



Developer	July 2015	Percent	August 2015	Percent	Change
Apache	482,624	48.26%	477,849	47.78%	-0.48
nginx	222,684	22.27%	226,105	22.61%	0.34
Microsoft	119,188	11.92%	117,526	11.75%	-0.17
Google	23,248	2.32%	22,717	2.27%	-0.05

Figure 1-3. Web server market share, top million sites, 2008–2015 (source: Netcraft)

While we don't have time or space to discuss all the popular web-related open source software projects that became the heart and soul of the Internet, it is worth noting that Linux and Apache formed the foundation of what was commonly termed the *LAMP stack*. The *M* stood for the vastly popular open source database MySQL, and *P* represented PHP, a popular scripting language for the Web that has only recently been eclipsed by the Node.js project (also an open source software project, and now a foundation as well).

Mobile Devices

Leaving the realm of servers and the web technologies that went along with them, we turn to the world of mobile devices. The explosion of the modern mobile device era, marked by the introduction of the *smartphone*, only dates back to 2007. That year saw two key events: the heavily anticipated iPhone launch with Apple's iOS, and the introduction of Google's Android OS for mobile devices. While both Android and iOS have their significant proponents, and debates continue to this day about which platform is "better," it is clear that as an open source project, Android has enabled a significant ecosystem of phones, tablets, and other devices across myriad

manufacturers. Due to this broad market, even though revenue numbers tend to favor Apple, active worldwide handset delivery numbers show Android in the lead (see Figure 1-4).

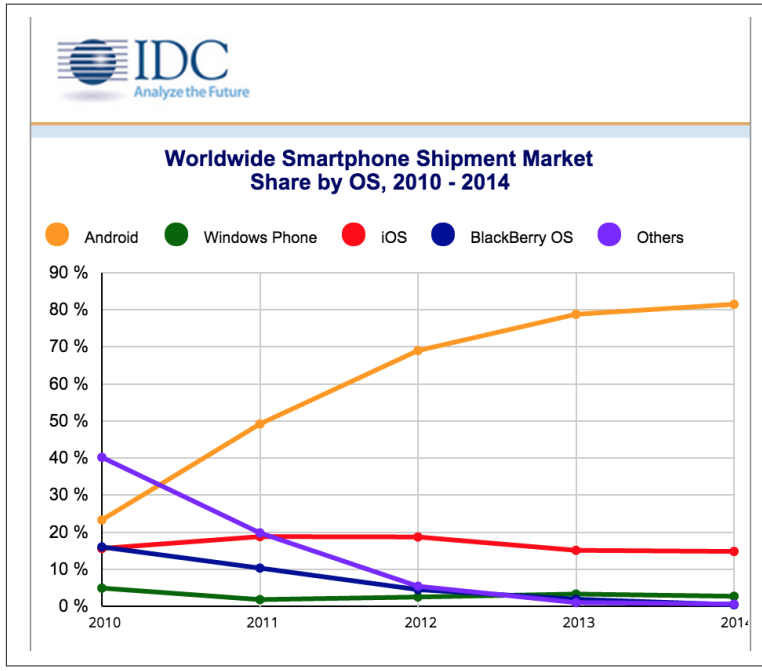


Figure 1-4. Worldwide smartphone OS market share, 2010–2014 (source: IDC data)

Given that Android enables the low-cost and entry-level market more favorably than the iOS platform, it is no surprise that finer-grained data shows a nearly order-of-magnitude difference between iOS and Android yearly shipments in major markets in India, China, and other developing nations, as noted most recently in the full-year 2014 data.

Virtualization

While software hypervisors existed long before the advent of VMware Workstation in 1999, many of them were part of significantly expensive enterprise servers from manufacturers such as IBM, Sun, and HP—systems that most engineers would never approach during their careers. However, when VMware Workstation appeared on the scene, what technologist doesn't remember the

wonder and excitement of seeing a virtual computer booting through a BIOS sequence inside a window on a laptop or PC? And for nearly a decade, virtualization was the hot topic: not only because of the ease of hosting physical workloads in virtual machines that were simple to back up, configure, and migrate, but as an entirely new way to pack larger numbers of isolated workloads onto the same physical hardware, leading to a major shift in data center operational models.

It wasn't long before the open source community also had offerings in the virtualization arena. The Xen hypervisor appeared first in 2003, offering a paravirtualization-based kernel feature for Linux; combined with the QEMU emulation software it has continued to grow in features and functionality, such as offering hypervisor capabilities to non-x86 architectures like POWER and, more recently, ARM. You might recognize one of the oldest public cloud offerings, Amazon Web Services (AWS), which has offered virtualized computing to end users since 2006. What you might not be aware of is that AWS runs its virtual machines atop the Xen hypervisor.

Also in the mid-2000s, an Israeli startup named Qumranet was working on its own hardware virtualization-based hypervisor named KVM, exploiting the Intel VT-x (or AMD-V) hardware-assisted features for virtualization. KVM was merged into the mainline Linux kernel in 2007, and Qumranet was acquired by Red Hat in 2008; KVM went on to become one of the most popular hypervisors supported across many Linux distributions and was the basis for several Linux enterprise virtualization products, such as Red Hat Enterprise Virtualization (RHEV) and IBM's PowerKVM product for the Linux-centric Open POWER hardware platform.

Cloud Computing

Given that software and hardware virtualization is *the* core technology that enables "cloud computing" to even exist, it provides a perfect segue to look at this most recent area of rapid innovation and significant market investment. All the major players in hardware and enterprise IT have created offerings or are significantly involved in the private, public, and hybrid cloud arenas.

While there are proprietary players in this space, what we are seeing today is myriad enabling open source software projects playing significant roles in the unfolding of cloud computing innovation. In

addition to pure open source projects, the lines are blurring, as we see traditionally proprietary players like Microsoft hosting Linux virtualization offerings in its Azure cloud, with an even more recent push to work upstream in the Docker open source project to bring container technology to Windows Server and the Azure cloud as well.

In essence, as Sam Ramji, executive director of the Cloud Foundry Foundation, stated recently: "Open source has won." It is difficult to envision any current cloud computing offering being devoid of some open source component, be it at the hypervisor or host operating system layer, or up at the application runtime layer, with open source projects like Node.js, PHP, Ruby, and Python as popular examples.

What we are seeing today is an open source renaissance, where much of the critical activity and innovation around the cloud is happening in and through open source communities and their respective foundations. Three of these communities are worth highlighting, as they have had significant impact on major IaaS and PaaS implementations from the largest IT enterprises. OpenStack, Cloud Foundry, and Docker all have substantial open source communities and are continuing to grow rapidly, with annual conferences boasting thousands of attendees, significant media coverage, and a broad range of partners and supporters from all the biggest IT enterprises. In Chapter 2 we will begin to the look at the introduction of the foundation model as a maturation point of open source, and how it has impacted both the communities mentioned previously and several historically large open source projects.

CHAPTER 2

Open Governance: The Foundation Model

Beyond Open Source

We've seen that open source is no longer a collective of independent or unaffiliated parties: the commercialization and popularization of open source has brought with it the investment and involvement of corporations and large enterprises. Along with that, however open source–savvy the participants are, there will obviously be potential conflicts between commercial and community interests in these projects.

> The intersection of open source and commercial interests raises questions about authority, authenticity, and culture.
>
> —Nathen Harvey, *Information Week*[1]

Three questions that Nathen Harvey asks in his *Information Week* article on the topic are: "Is the project driven by the commercial sponsor or outside contributors? Will commercial interests trump the wishes of the community? How and where do you draw lines between a commercial entity and the open source community?"

These are critical questions to answer, and many of them can be resolved through the process of open governance via the foundation model. First, it will be helpful to understand the history and rise of foundations in the open source software world.

1 From "Three Pillars of Open Source Governance."

Rise of the Foundations

Let's look at a few of the more significant foundations and their roles in specific communities. By taking a quick walkthrough of these foundations we can better understand the way in which specific communities developed their shared visions via the open foundation model.

Apache Software Foundation

The Apache Software Foundation (ASF) was established in 1999 and, at the time, was mainly focused on coordinating the development, funding, and governance of the Apache HTTP web server project. Today, it is one of the most widely known and successful open foundations for hosting open source software projects, with more than 300 such projects under its domain. The ASF paved the way for many other open source hosting and collaboration efforts by defining the legal and collaborative frameworks that many other foundations have emulated to this day. For example, the Apache License, under which all Apache projects are distributed, is one of the most popular and accepted open source licenses in use today, even well beyond projects directly hosted under the ASF. Though the ASF started with a sole focus on the Apache web server project, it has branched out into a broad range of other technologies, including programming languages, cloud computing, and even office productivity tooling.

The ASF operates as a meritocracy and conducts all of its business and project work in the open via popular social technologies, such as public mailing lists, wikis, and source code repositories. While several projects developed under the ASF have become very popular, and might even be seen as de facto standards for certain technologies, the ASF is not a standards body and does not create "standards" such as those organizations like the W3C produce.

Linux Foundation

The Linux Foundation was founded in 2007 through a merger of the Open Source Development Labs (OSDL) and the Free Standards Group (FSG), with the express purpose of providing a vendor-neutral foundation that would support the development and enhancement of the Linux operating system and related technologies. According to its website:

> The Linux Foundation protects and promotes the ideals of freedom and generous collaboration established through the development of Linux, and shares these ideals to power any endeavor aiming to make the future a better place in which to live.

One of the early intentions of the Linux Foundation was to provide an independent entity in which Linux creator Linus Torvalds could work without dependence on a commercial entity that might seek to provide undue influence on the direction of Linux kernel development priorities. This provision has continued through today, with other key Linux community maintainers like Greg Kroah-Hartman also being employed at the Linux Foundation. Beyond this safe harbor for key Linux leaders, the Linux Foundation promotes Linux through worldwide event management, protects the key Linux trademark and handles legal and license issues, and helps standardize key Linux interfaces through work such as the Linux Standard Base working group.

Linux Foundation collaborative projects

Similar to the ASF, the Linux Foundation has matured through the years to become a collaborative umbrella for other open source projects that are related to Linux, but not necessarily specific to Linux operating system development. Through its Collaborative projects initiative, existing Linux Foundation processes, administrative support, and governance procedures can be harnessed to quickly and efficiently charter new collaborative endeavors. In recent years the growing list of LF Collaborative Projects has included everything from the Open Mainframe Project to Automotive Linux to very recent cloud computing projects such as the Cloud Foundry Foundation, OpenDaylight, OPNFV, the Open Container Initiative, and the Cloud Native Computing Foundation.

To get insight into how these collaborative projects are being used specifically in the cloud computing world, let's take a brief look at four of the most recently chartered foundations under the Linux Foundation's Collaborative Projects umbrella.

Cloud Foundry Foundation. Cloud Foundry is an open source project providing a PaaS (Platform-as-a-Service) environment for efficient developer application delivery across a wide range of runtime environments, from Java, to Node.js, to Python, PHP, Ruby, and Go-based applications. Cloud Foundry was originally developed by

VMware but has since been jointly owned by VMware, EMC, and General Electric under a new company, Pivotal Software.

While Cloud Foundry under Pivotal was an open source project, with Pivotal providing both free and commercial offerings based on the open source codebase, Pivotal clearly had significant control over the direction of the project and community. To address this single-vendor control point, in December 2014 the Cloud Foundry Foundation was announced under the umbrella of the Linux Foundation Collaborative Project. This new organization now ensures that no one company dominates the leadership responsibilities of the project and aligns Cloud Foundry with a true open governance model.

Open Container Initiative. While we will talk more about the Open Container Initiative (OCI) in Chapter 3 when we discuss collaboration in the cloud, it is useful to note it here as one of the key open cloud initiatives formed under the umbrella of the Linux Foundation Collaborative Projects. In June 2015, the OCI was formed out of a desire to standardize and harmonize two competing runtime layers for the containerization of applications. It is an understatement to say that Linux containers have been an extremely hot topic in the last few years, with Docker dominating the conversation as the go-to implementation for an application container runtime and ecosystem. It is also worth noting that Docker appeared on the scene very recently, even though some of the core Linux technologies that make up what we call a "container" have existed for over a decade! Given the continuing hype cycle around containers, Docker is not without its detractors and competitors, and in December 2014 CoreOS announced Rocket, a competing container runtime. The OCI has been set up as a vehicle for the harmonization of these two specifications and implementations, with Docker contributing *libcontainer*, its core container runtime component, and a new user runtime named *runC*, an implementation of the specification, to the OCI's governance and control.

In many ways, the OCI is still in its formative stages, but both CoreOS and Docker (as well as Google, Red Hat, IBM, Huawei, and others) are among the initial founding members, with the goal being to create a portable and standardized runtime specification with a reference implementation available for consumption by innovative container ecosystems like Docker and Rocket.

Node.js Foundation. Invented in 2009 by Ryan Dahl and a team of engineers at Joyent, Node.js has proven an increasingly popular JavaScript-based server framework for web application development. Node.js was an open source project from the beginning, licensed under the MIT license. Joyent guided its development for several years, taking it from a Linux-only runtime to on that supports Microsoft Windows and Apple's OS X, and other CPU architectures like IBM POWER and z. In late 2014, due to internal differences over Joyent's governance, a forked open governance–based project named io.js was announced, threatening the future of a single Node.js runtime and release cycle for the significantly large user community.

At the annual Node.js conference in February 2015 the intent to create a vendor-neutral foundation was announced, and by early summer, the two projects had combined under the new Linux Foundation Collaborative Project–governed Node.js Foundation. At this point, Node.js looks to be a success story of the open governance model as implemented via a vendor-neutral foundation. Similar to many of the other foundations we have discussed, it will use a business (board) committee in combination with a technical steering committee, with the latter being run as a meritocracy for technical decision making.

With 6 platinum founders, including the technology's creator, Joyent, and 16 other gold and silver foundation members, the Node.js foundation is forging ahead with the same momentum as the increasingly popular technology for which it guides development.

Cloud Native Computing Foundation. Whereas the OCI, discussed earlier, covers the low-level runtime specification for application containerization, many of the key cloud players realized the need for standardization beyond the base runtime layer provided by the OCI's work. While agreement on the basic management of a single container runtime is critical, the need to extend this industry-wide agreement to higher-level management constructs became evident. In August 2015, again under the Linux Foundation Collaborative Projects umbrella, the Cloud Native Computing Foundation (CNCF) was formed. As of our publishing date, the exact scope of the CNCF's work is still being finalized, but a general focus will be on the orchestration, distribution, discovery, and lifecycle management of clusters of containers within a data center. The collection of

these technologies has been referred to as a "Datacenter Operating System" (DCOS).

Like the OCI, the CNCF will be taking a code *plus* specification approach, developing the specifications of the DCOS architecture at the same time as the implementations. Already, Google's Kubernetes and Apache Mesos projects are being discussed as potential implementations of the CNCF's specifications.

Similar to the Node.js Foundation, the CNCF started with a broad range of key cloud industry players: at the time of writing a total of 22 companies support the CNCF, including large enterprises such as IBM, Intel, Huawei, Google, and AT&T. Interestingly, as we begin to look at cross-collaboration in the cloud due to open source and open governance, the supporting members also include Docker, Cloud Foundry, Weaveworks, and CoreOS—all open source projects in their own right that are interested in collaborating on this key orchestration and management topic for the future of the cloud. We'll return to that discussion in the following chapter.

OpenStack Foundation

OpenStack was announced in 2010 as a joint project between NASA and Rackspace to create a programmable, API-centric IaaS (Infrastructure-as-a-Service) layer for data center infrastructure management. By 2012 several other key vendors had joined the open source initiative, and together they created the OpenStack Foundation, which now operates the legal, technical, and administrative governance of the OpenStack development project.

The OpenStack project originally focused on the management of compute, storage, and networking resources, but has grown to include a wide range of IaaS-related projects, each with a specific code name; they begin life as incubated projects and then are potentially promoted to the official OpenStack semiannual release. Currently, 16 unique subcomponents are scheduled to ship in the Liberty 2015 release of OpenStack.

The foundation itself has grown to encompass a broad range of operator, user, and developer communities with a very strong governance model that is codified in its bylaws, which guide its governance and technical development processes, overseen by a board of directors and the technical committee. OpenStack now has 8 plati-

num sponsors (all with representatives on the board of directors), 17 gold sponsors, and over 120 corporate sponsors.

Without this formal governance and development process, it would be unexpected for competing cloud providers to work together on an IaaS layer and API. But, given the foundation structure, companies such as HP, Rackspace, IBM, Intel, and others are collaborating together on the core technology and innovating and differentiating in their own product platforms based on OpenStack. We'll look more closely at this cooperation and "co-opetition" model in following sections.

Other Open Source Foundations

We don't have the time or space to cover other key open source foundations that exist to support open governance and related processes around commercial cooperation in open source. However, it is worth mentioning that many foundations have been critical in ensuring the overall health of open source software as it stands today. Foundations such as the Free Software Foundation (FSF), Creative Commons, the Eclipse Foundation, the Internet Systems Consortium (ISC), the Mozilla Foundation, the Open Source Initiative (OSI), and the Software Freedom Law Center (SFLC), among others, have all played a role in providing an even playing field for commercial and independent interests alike to cooperate in the development of open source software projects and initiatives.

The "Other" Open Source: Open Standards

Before we wrap up our discussion of open source and open governance, it is worth mentioning that outside the current flood of commercial involvement in open source software projects, many industries have used non-source-code–based attempts to achieve similar vendor-neutral collaboration and co-opetition through the use of standards bodies.

Ignoring the software industry for a moment and simply thinking about everyday items you find around the house, you'll uncover numerous examples where corporations, and in some cases government agencies, have joined forces to create a standardized design or component with the intent that this will be beneficial for all parties —both producers and consumers. These standardized parts or designs, like a USB port or national electrical plug type, enable a cer-

tain degree of interoperability, portability, and reuse between manufacturer products. Having an agreed-upon standard still allows innovation and distinctive features to be developed on top of that base standard; each manufacturer or producer still has the latitude to differentiate itself while benefiting from a common standard.

Within the software world, we see this co-opetition occurring as well. For example, as previously mentioned, the Apache web server saw huge success through its collaborative open source project. But the Apache project wasn't started with the purpose of creating a shared standard on the core technology of a web server; rather, it was about using the source code as a basis for collaboration. Other entities, such as the W3C[2] (established in 1994 by Tim Berners-Lee, widely viewed as the creator of the World Wide Web) and the IETF, [3] were established to develop the protocol standards that have made the Internet the huge success it is today, and without these projects like Apache would never have even had a chance to succeed.

These organizations, and the companies that made up their membership, understood that without some degree of agreement on and commonality across the basic underpinnings of how computers and people communicate, the ubiquitous use of the Internet would simply not be possible. For example, imagine the challenge of the Web without an HTML standard, with each website implementing its own language for described page layout! The Internet—and specifically the Web, as a source of nearly limitless, easily accessible information—would not exist as we know it today without the core standards put in place years ago.

While open standards bodies still play a critical role in parts of the industry even today, we are seeing a shift from pure "on-paper standards" to code as standards, or specifications with reference implementations, like the Open Container Initiative discussed in the prior section. Given this shift, we are finding that the open governance foundation model of overseeing open source code projects is a more compelling model for future work. Of course, future cloud

2 "The World Wide Web Consortium (W3C) is an international community that develops open standards to ensure the long-term growth of the Web." (From http://www.w3.org.)

3 Internet Engineering Task Force. "The mission of the IETF is to make the Internet work better by producing high quality, relevant technical documents that influence the way people design, use, and manage the Internet." (From https://www.ietf.org.)

computing activity may warrant new open standards that will lead to open source implementations, but we believe that will be a case-by-case decision led by vendor-neutral industry consortiums to again provide a level playing field for both open standards and open source software.

Open Governance: Critical for Cooperation

We began this chapter by looking at critical questions related to commercial involvement in open source software projects. As we've examined specific open governance implementations via the foundation model, we've noted that open governance practices answer these questions about influence, ownership, leadership, and decision making in open source. While every project will have its growing pains as a mix of independent and commercial contributors join to participate, we believe that open governance via vendor-neutral foundations, built on meritocracy-style technical development practices, will enable the continued rise of open source as the de facto development mode for both commercial and non-commercial entities alike.

In the next chapter, we want to look more practically at the pros and cons of how communities operate and give examples and guidance for those looking to jump on board with the open source revolution, including those who might be planning on starting new collaborative projects in the future.

CHAPTER 3

Collaborating on the Open Cloud

Given our prior discussions of the popularization and commercialization of open source, we know that *both* collaboration and competition will continue to coexist for the foreseeable future in open source projects. We've looked at a set of foundations that have succeeded in bringing about a vendor-neutral playing field in which multiple corporate and independent members can operate under the same set of meritocracy-based rules for development, technical planning, and decision making in a project. In this chapter, we look more completely at the key indicators of success in cloud computing open source projects. We will also give guidance for future projects based on what we find when we dig into what has been shown to both work and not work for historic and current open source projects and standards bodies. First, let's summarize what we saw regarding governance models in the last chapter.

Successful Collaboration Through Open Governance

It's worth trying our best to answer the question, "What makes a good governance model?" Looking at success in past and current projects—for example, the Apache Software Foundation or the W3C—a key aspect we find in common is the measure of "openness" within the project's culture: both its human organization and its administrative and development processes. "Openness" in this context can take different forms, but examples include:

- Does the community operate in such a way that anyone—even people outside the community—can follow community discussions and progress? For example, are all of the documents (meeting minutes, lists of issues, future work items, and so on) kept on publicly accessible websites, easily found by and visible to all interested parties?
- Do they welcome input from non-community members? Are there mechanisms in place for members as well as non-members to provide feedback, ask questions, and suggest changes?
- Do they have an IP policy in place that encourages the sharing of ideas? Clearly, certain open source projects have licenses that make it a challenge for companies to leverage the source code in commercial products. Conversely, if the IP policy in use doesn't require contributions to be unencumbered and freely reusable by the community (without, for example, paying a royalty fee), then that will strictly limit the exchange of ideas and, therefore, community success.

While certain governance models might include some level of hierarchy in the membership (based on annual investment, for example), there must be a fair process in place for all community members to have an equal chance to rise through that hierarchy. For example, a "pay to play" model where a company can gain a key leadership role by simply writing a check hurts the perceived objectivity of the organization. Typically, for technical community leadership, a merit/contribution-based model works best. In many of the open source projects we've covered, only after submitting a significant number of meaningful code contributions is a contributor considered for a code committer, reviewer, or maintainer role.

We also note that it is of utmost importance that projects have vibrant, open, and welcoming communities. Successful projects leverage all of the aspects we've previously discussed to create an environment where new and existing members of the community are treated equally. How "old-timers" treat newcomers is often a sign of the openness of a community. Taking the time to answer questions from "newbies" and help guide them through any issues related to developer setup, initial change submissions, and the like helps grow the community and make it more successful, as it encourages more contributors and therefore a broader range of expertise and contributions. As a project matures, we note that existing developers can

tend to move on to newer assignments, and therefore it's critical for the ongoing health and success of an open source project to have a steady stream of new contributors. If interested newcomers feel discouraged or ignored, they'll look for other, more welcoming venues in which to spend their time and efforts.

Case Study: Closed Standards and Private APIs

In this section, we'll briefly look at two counterexamples in our tour of open source and open standards–based governance models that highlight the value of true open collaboration in cloud software projects.

Closed Standards: Cloud Infrastructure Management Interface

We've spent a fair amount of time discussing open source projects and foundations, while only briefly mentioning some non-code-based collaborative efforts, such as the W3C's work on "paper standards." One such effort within cloud computing is the Cloud Infrastructure Management Interface (CIMI). This set of cloud computing specifications is being developed under the DMTF,[1] with the intent to promote a set of IaaS APIs for managing virtual cloud resources.

The work of the CIMI represents the traditional standards development method used by the Open Group and other historic consortiums. While standards from these bodies, like POSIX, are still valuable today, for the most part their work is done in private, with paying membership, and not open to public scrutiny. There is no visibility to their discussions, mailing lists, issues, or future plans. While several of the CIMI specifications have involvement from key industry players—for example, IBM is involved in the CADF specification work, the results of which have been applied to the OpenStack auditing and logging features that now use the standardized CADF message format—they are generally focused on a "specification first" approach. While sometimes valuable, these specifications are being developed in the absence of real-world scenarios and implementa-

1 Distributed Management Task Force, Inc. "DMTF standards support implementations that enable the management of diverse traditional and emerging technologies, including cloud, virtualization, network, and infrastructure." (From http://www.dmtf.org.)

tions, and are potentially missing valuable input from non-paying parties or open communities with broad expertise: the "many eyes" phenomenon that has been written about generally in open source.

Unfortunately, this typically means that a reference implementation is developed strictly for the purposes of testing the specification and is not linked with real-world software systems. This may mean that the specification—created in a vacuum of sorts—will not even meet the critical end user or operator needs, which are more readily apparent in an open and transparent community development effort around either open standards or an open source implementation.

While the work on the CIMI specifications is continuing, it's not clear how much adoption they will have in the broader cloud community. Because of the nature of how the DMTF creates specifications, and the lack of a closed feedback loop with an open community, the cloud industry as a whole appears to have moved toward other venues for standardization on IaaS-layer APIs, such as OpenStack.

Private APIs: Eucalyptus

Eucalyptus is an open source project first released in 2008. It was one of the first IaaS open source projects developed and, for a time, had growing momentum among some segments of the cloud computing community. However, in recent years we have seen Eucalyptus decrease in popularity, despite its being acquired by HP in 2014. While we won't venture to speculate on all the reasons why a project like Eucalyptus hasn't seen the broad adoption that more recent IaaS projects have seen, we do have a few observations that are worth noting.

First, Eucalyptus was specifically written to be compatible with Amazon's IaaS cloud offering, Amazon Web Services —it was a direct implementation of Amazon's cloud APIs. Given AWS's market-leading position in the public cloud market, it most likely seemed a good choice to provide an API-compatible alternative to an Amazon-hosted solution. This API conformance would allow for a private and/or on-premise cloud implementation via Eucalyptus that would be entirely compatible for users of the market-leading AWS public cloud. However, we note two problems with this approach. First, as an AWS-compatible solution, Eucalyptus had to defer its design, feature set, and even potential success to Amazon's

AWS cloud—and Amazon's relentless march toward more features (and therefore more APIs) meant that Eucalyptus would be in a near-constant "catch-up" mode trying to match feature and API changes made at the whim of Amazon. Eucalyptus effectively handed Amazon control and influence over those decisions.

Additionally, and maybe more importantly, Amazon's API and cloud infrastructure management are closed and proprietary implementations. Amazon provides no licensing guidance for its APIs, leaving it up to potential differences of legal opinion whether other companies have the freedom to implement the AWS API set. Given that Eucalyptus relies on the ability to provide AWS APIs as its core and only API offering to end users, this is a potentially risky proposition. While we have seen other open source IaaS projects provide AWS-compatible APIs as well, these are typically made available for compatibility and migration and are not the core API set provided for the end user.

In summary, we should note that as the case of Eucalyptus illustrates, being an open source project does not overcome the fact that there can be only limited collaborative activity and nearly zero true open governance around an implementation of a closed, single-vendor IaaS API. Whatever other factors are impacting Eucalyptus's uptake in the cloud computing world, we can at least surmise that an open source cloud *requires* open collaboration and open governance across the entire ecosystem—from API definition to implementation—to provide a more welcoming, vendor-neutral community that can then attract contributions from a larger pool of individual and corporate players to generate momentum.

Case Study: Open Source Builds Open Clouds

After looking at those two counterexamples to our open collaboration and governance model, we'll now revisit three major cloud open source projects we briefly discussed from a foundation perspective in Chapter 2. All three of these projects—OpenStack, Cloud Foundry, and Docker—have open ecosystems with broad community participation, fully implemented or under-development open governance, and increasing industry momentum that is impacting real-world, production-ready cloud computing offerings across the spectrum from startups to large enterprises.

OpenStack

As we noted in Chapter 2, OpenStack is a large and fast-growing open source project aimed, primarily, at providing a comprehensive API and implementation for the IaaS layer of the cloud stack; it is focused mainly on compute, storage, and network resource management. The OpenStack Foundation has been formed to take on the responsibility for governance, trademark and legal oversight, administration, and promotion of the OpenStack project.

It is worth noting, though, that while OpenStack has broad industry support and is used by many top-tier cloud providers, its open source code implementation of the API specifications (which start out life in the OpenStack model as "blueprints") is not codified with a traditional "paper standard" specification. Therefore, additional bylaw changes as well as new projects have grown up within the OpenStack Foundation to combat the potential for lack of interoperability between vendor implementations. The RefStack community project and DefCore committee within OpenStack are remedying this situation by providing a test suite and required "core" software code implementations that will need validation if vendors wish to use OpenStack marks and be certified as compatible. While these kinds of bumps in the road can be expected in such a large and diverse community, the OpenStack Foundation governance and meritocratic development models are providing a solid framework for continued collaboration and the growth of the community in positive directions.

OpenStack is still young in many ways, but with OpenStack-powered clouds and offerings available from significant players like IBM, HP, Rackspace, Huawei, and Cisco (Piston), among others, the momentum is definitely growing for OpenStack to play a vital role in open cloud collaboration for years to come.

Cloud Foundry

The Cloud Foundry (CF) open source project provides a PaaS environment to its users, focused on efficiently providing application developers with a deployment framework that handles runtime parameterization, scalability, and application lifecycle needs, such as monitoring and autorestart. CF also automatically manages the load balancing and routing of requests from end users of the application as needed, removing many of the tedious tasks normally associated

with application management for both the developer and operator alike.

As noted in Chapter 2, Pivotal has turned over governance of the CF project to the Cloud Foundry Foundation, and it has been operating with open governance and vendor-neutral meritocracy-based development processes since late 2014.

While Cloud Foundry has been an open source project from the beginning, the addition of open governance has provided a healthy ecosystem for continued vendor adoption and community growth to a broad set of cloud industry participants. With Cloud Foundry–based PaaS offerings from IBM, HP, ActiveState, Pivotal, and CenturyLink now available, the momentum for Cloud Foundry continues to grow, and the young but solid governance structure provided via its open foundation has set up a bright future for Cloud Foundry and for collaboration around PaaS solutions across the industry.

Docker

Docker is one of the newest open source projects to take cloud computing by storm. At its core, Docker aims to take existing container technologies—which are sometimes a confusing array of capabilities, or even downright cryptic Linux kernel functionality—and provide a simple and clear user experience to "build, ship, and run" application code anywhere, potentially in place of traditional virtual machine or even PaaS use cases. And if the last two years have anything to say, Docker has succeeded in this quest in a very big way. Recently, Enterprise Technology Research surveyed 685 enterprise CIOs on their intention to spend money on Docker-related technology in the next year. The surprising result was that 97% specified an intent to do so—the highest intent score that ETR had ever seen.[2] As another proof point of Docker's success and influence, Amazon recently agreed to support Docker's APIs via its own AWS EC2-based container service. While Amazon does support many traditionally developed standards across its AWS platform, it is rare for it to adopt another's nonstandard API definition. This was clearly done in recognition of Docker's leadership position in the containerization space—similar to the way we have noted other projects

2 Thomas DelVecchio, "Docker Scores the Best Ever NET Score in ETR History," April 17, 2015.

providing AWS IaaS API-compatible implementations due to Amazon's own dominant market position.

Similar to the beginnings of other cloud open source communities we've discussed, the Docker open source project is currently overseen by a single-vendor commercial entity of the same name. While Docker, Inc. does hold key roles and maintain leadership of the open source project, we have experienced that the Docker community has many of the same positive aspects that we've discussed as beneficial throughout this book. Almost all of Docker's open source community work, planning, and discussions take place in public forums. The Docker, Inc. employees as well as open source community members are extremely good at providing a very welcoming experience to new members; in fact, our experience is that they (more than most other open source projects) go out of their way to encourage new members to join, no matter their experience level or level of project knowledge. While it's not ideal that the open source project has single-vendor control, as we've seen many times before, this is fairly normal for a young project. We believe what happens in the next cycle of Docker's maturity will determine the long-term success of the project and its governance and oversight.

With Docker's success has come competitive pressure from other cloud industry players seeking to also have a voice in the future of the white-hot focus on containers as the future of cloud computing application delivery. With that pressure has also come some community fracturing: most notably when CoreOS, a long-time Docker proponent, announced its own container runtime implementation, named "Rocket," alongside a container runtime specification initiative, "appc," in December 2014. While this publicly exposed friction and community rumblings have caused some dark clouds (hopefully only temporarily) to gather over the Docker project, the initial response has been a positive first step toward vendor-neutral open governance for container technology in general, and core pieces of the Docker runtime specifically.

As of June 2015, Docker has contributed the core of its container runtime codebase—a subproject named "libcontainer" as well as a new runtime interface called "runC"—to the Open Container Initiative, which we will discuss in more detail in the next section. Over time, we expect a continued maturing of the open governance and open collaboration around containers, which will allow for innovation and co-opetition between commercial and open source con-

tainer product offerings, but with the interoperability and transportability that relieves the fear of vendor lock-in for consumers of this currently very hot technology.

Case Study: Open Foundations Extending Cloud Collaboration

While all of the open source cloud projects we've looked at are interesting and valuable in their own right, what we are seeing on the horizon, and expect will become normative for the future, is the creation of collaborative open foundations that span multiple open source projects. These cloud computing foundations will encompass specific technology areas to allow standardized interfaces and definitions to open up cross-project collaboration to an even greater degree than we are seeing today. These are exciting times for solving the next generation of cloud challenges, and we want to highlight a few recent foundations that we feel fit this new era of open cloud collaboration.

Open Container Initiative

As previously mentioned, the Open Container Initiative (OCI) is a project being run under the Linux Foundation that was formed using Docker's libcontainer component as a starting point implementation for standardizing the container runtime model. In addition to having a reference starting point with libcontainer, the OCI is chartered to develop a specification that will harmonize the work that CoreOS has done on the appc spec and Rocket with the Docker de facto implementation. While at this time the OCI project is still in the process of drafting a final approved charter and agreeing on project scope, it is expected to at least standardize both the definition of the container's packaging or bundle format and the runtime specification and APIs that determine how a container is managed through its lifecycle. The bundle or packaging represents the contents of the container's filesystem and all runtime and configuration metadata needed for a runtime to execute it. The lifecycle definition will define how to manage the container's runtime via starting, stopping, checkpointing, and restoring the container.

The OCI is a good example of the next generation of standards development and cross-project collaboration. As previously discussed, in the past there have been standards organizations that

developed "paper standards" and then asked for multiple implementations to test those specifications. Conversely, there have been plenty of open source projects that focused only on the code implementation and whose APIs, given enough popularity, eventually become de facto standards without any specification, interoperability testing, or joint agreement between all interested parties.

With the OCI there will be the attempt to do both: the specification, or standard, will be developed in concert with the reference open source implementation. While the model isn't necessarily new, Docker has agreed to use that reference implementation as part of Docker itself. In addition to Docker consuming the community-defined reference implementation, we have also seen the Cloud Foundry development community propose to use the reference implementation, runC, as their container runtime within the Cloud Foundry application framework. These early highlights from the OCI show that this reference implementation will not only be an accurate representation of the specification but also will immediately be used and tested in real-world scenarios, with actual customer experience to help ensure that the specification is directly addressing the requirements of the cloud community. Additionally, we're already seeing multiple implementations of the OCI specifications under development. This ensures that no one particular implementation is codified within the specification unnecessarily. This focused attention on linking the standard with a real-world codebase, developed in concert with multiple cloud vendors, is a natural next step for the standardization process and aligns well with the open governance models we've noted as the cornerstone of successful open source projects.

Cloud Native Computing Foundation

Not long after the OCI was formed in June 2015, many of the key cloud players realized the need for standardization beyond the base runtime initiative the OCI was focused on providing. While agreement on the core management of a single container is critical, the need to extend this contract to higher-level container management constructs became evident. In August 2015, within the scope of the Linux Foundation Collaboration Project umbrella, the Cloud Native Computing Foundation (CNCF) was formed. While, as of this writing, the exact scope of the CNCF's work is still being finalized, it appears they will build on the foundation of the OCI's work and

focus on standardizing the orchestration, distribution, discovery, and lifecycle management of clusters of containers within a data center.

While we can't state what the exact scope of the CNCF's work will be at this time, we are again hopeful that with such a broad array of key industry players, significant collaboration will occur to provide standardized answers for some of the next challenges that are a step removed from the foundation laid by OCI. Specifically, at this higher orchestration and lifecycle layer, we would want to see commonality around distribution, discovery, and management features, allowing for the development of open and hybrid cloud solutions that interoperate between providers.

Playing Your Part in the Open Cloud

We've looked at several case studies of open source, open governance, and the foundation model and seen the value and future of true openness as the path to collaboration with innovation and coopetition for cloud technologies. We've also noted that among major cloud initiatives and projects, there is a growing sense that cross-collaboration with other major projects is the way forward to solve the cloud computing challenges looming on the horizon.

We can also see that the lines are blurring between operator, developer, producer, and consumer, and that the nature of the code, community, and culture of open source is shifting, transcending traditionally designated roles. This is leading to an era where truly anyone can be an open source developer and contribute to projects that interest them, whether by shoring up documentation as an avid end user or providing proposals for new features as a potential innovator within the project's field.

For interested end users, this means no more standing on the sidelines as a pure consumer: getting involved in projects of interest allows them to have a voice in the future direction of those projects, based on their own needs. For companies producing cloud platforms or operating cloud offerings, investing in open source can both benefit and highly accelerate time to market for required capabilities, as well as allowing them to provide their own niche expertise to a broader open community of interested parties. Getting involved in the open governance foundations around key cloud projects also helps provide a continued level playing field, as broader

participation provides more voices to perpetuate vendor-neutral decision making.

If you are considering open sourcing an in-house technology, or creating a new community project guided by either independent or corporate leadership, remember that intentionally choosing openness in the entire ecosystem of your project highly correlates to the long-term viability of that project, as we've seen. You will need to allow other parties the ability to have leadership roles, guided by a healthy open governance model and meritocracy-based technical development practices, and this will most likely be difficult at first—who wants to cede control of a personally birthed idea or project? However, as we have shown, this path is most beneficial to true collaboration in the cloud.

Summary

We believe that open source software projects governed by healthy open governance principles, often delivered practically through vendor-neutral foundations, is the future of successful cloud computing technology. Already we have seen significant projects like OpenStack, Cloud Foundry, Docker, and the foundations we have discussed around each of these gaining momentum, partnerships, corporate sponsorship, and participation, leading to viable production cloud offerings coming from an array of vendors, both small and large. These open source and openly governed projects both allow for broad community collaboration as well as unique innovation, enabling many traditional enterprises and startups to generate revenue streams from cloud offerings based on open source technology.

Additionally, we believe the next phase of collaboration in the open cloud is for growing cross-project collaboration through umbrella foundations looking to standardize key pieces of both the low-level and high-level orchestration and management components of cloud computing. What most customers need is not a one-size-fits-all approach to IaaS or PaaS, but rather a holistic approach that covers potentially multiple open source projects and offerings. Having standardized interfaces for orchestration, cluster management, and distribution/deployment across multiple cloud infrastructure types—for example, VMs and containers—is the next horizon, what we believe will take cloud computing to the next level. When common

and interoperable implementations of these key concepts are collaborated on across many vendors and many related open source projects, even more potential for new revenue streams, new offerings, and new niche vendors will be realized.

We believe the future of the open cloud is that it will be *Open by Design*.

About the Authors

Phil Estes works as a senior technical staff member within the IBM Cloud Open Technologies team, currently representing IBM in the open source Docker community as a core maintainer. Phil also works together with IBM product teams and customer accounts on applying cloud open source technologies to products, solutions, and IT projects. Phil's broader team works upstream in OpenStack, Cloud Foundry, Docker, and other key cloud open source projects.

Prior to his work with the Open Cloud team, Phil was the chief architect in IBM's Linux Technology Center for embedded Linux, and he has deep expertise in Linux operating system packaging and design. Phil has also worked closely with IBM product and legal teams to provide expertise on technical and legal issues around open source licensing, redistribution, and open source use within commercial products.

Phil holds a BS in Computer Engineering from Florida Tech and received an MS in Software Engineering from The University of Texas at Austin. He currently resides in beautiful central Virginia with his wife and five children, as well as a dog, a rabbit, and two parakeets.

Doug Davis works in the Cloud, Open Source, and Standards division of IBM. He has been working on open source and standards for over 15 years and has been involved in many of the more popular initiatives—such as Apache SOAP and Axis, most of the W3C and OASIS standards around Web Services/SOAP, OpenStack, CloudFoundry, and most recently Docker, OCI, and CNCF. He founded an interoperability consortium around Web Services, called WSTF, and hosts a website that is used by several organizations to manage their real-time collaborative discussions.