LINUX FOUNDATION

A $5 Billion Value:

Estimating the Total Development Cost of Linux Foundation's Collaborative Projects

By Jeff Licquia and Amanda McPherson
A Linux Foundation publication

www.linuxfoundation.org

In the past ten years, open source has taken over the software industry. As the speed and requirements of innovation are changing, technology companies have realized that to keep pace and be cost effective they must leverage external R&D in the form of open source projects.

Open source has many benefits: from increased interoperability to reduced costs as multiple vendors and users leverage shared resources for non-differentiated parts of the stack. Companies using open source are building products faster and more efficiently; and once the code is part of a vibrant open source project, the value through continued community support is multiplied. For infrastructure especially, large-scale open source development has become the *de facto* way to develop software.

Since 2008, the Linux Foundation has worked with the world's leading technology companies and most talented developers to host large-scale open source projects across multiple segments of the technology industry. Our mission is to adapt the principles and practices that have made Linux so successful and offer them to any endeavor working to solve complex problems.

Linux Foundation collaborative projects span almost every area of the technology stack. If you want to understand the future of technology, Linux Foundation projects will give you a good indication. These projects include:

ALLSEEN ALLIANCE IoTivity	**AllSeen Alliance** and **IoTivity,** building open platforms for Internet of Things (IoT) among others.
AUTOMOTIVE GRADE LINUX	**Automotive Grade Linux,** building the next generation platform for automotive In-Vehicle Infotainment (IVI)
CLOUD FOUNDRY	**Cloud Foundry Foundation,** the open standard cloud native application platform
CLOUD NATIVE COMPUTING FOUNDATION	**Cloud Native Computing Foundation,** building a lightly coupled stack for modern, cloud native applications leveraging containers

CodeAurora FORUM	**Code Aurora Forum,** providing the tested code needed to bring innovative open source-based products to market in the mobile industry
CORE INFRASTRUCTURE INITIATIVE	**Core Infrastructure Initiative,** helping critical, under-resourced open source projects
Dronecode	**Dronecode,** for non-military unmanned aerial vehicles
Let's Encrypt	**Let's Encrypt,** creating an accessible SSL/TLS certificate authority
node FOUNDATION	**Node.js Foundation,** and a JavaScript platform for applications
OPEN CONTAINER PROJECT	**Open Container Initiative,** establishing a standard specification and runtime reference implementation for application containers
OPNFV	**Open Platform for NFV,** for Network Functions Virtualization (NFV)
OPEN DAYLIGHT	**OpenDaylight,** for Software-defined Networking (SDN)
R consortium	**R Consortium,** for building an ecosystem supporting the R language used in data science
Xen Project · OPEN VIRTUALIZATION ALLIANCE	**Xen Project** and **Open Virtualization Alliance,** building virtual machine technologies and ecosystems
yocto PROJECT · TIZEN	**Yocto Project** and **Tizen,** building an OS and deployment workflow for embedded systems

There are well over 500 corporate members and thousands of developers collaborating on these projects.

The rise of open source projects and foundations has been much discussed, but the code investment underpinning them has not been analyzed in much detail. Many of the Linux Foundation's collaborative projects started with significant code donations from companies or existing projects. Once they became a fully open and collaborative project with neutral governance, many companies and individuals engage in development. Because of that, there is no single source for cost estimates of how much it would take to develop the technology or an understanding of how much value these projects actually deliver to the industry.

This paper attempts to answer two questions:

1. What would be the monetary cost of rebuilding or developing the software residing in The Linux Foundation's collaborative projects if an organization had to create it from scratch? What R&D value are the people who use these projects receiving?
2. What is the value in collaboration outside of this cost that is gained via commercial companies shipping this open code in products (and then working within the projects to improve and advance the code)? In short, what is the ecosystem accelerant inherent in these projects?

In 2002, David A. Wheeler published a well-regarded study that examined the Software Lines of Code (SLOC) present in a typical Linux distribution (Red Hat Linux 7.1)[1]. He concluded—as we did—that SLOC is the most practical method to determine open source software value since it focuses on the end result and not on per-company or per-developer estimates. Using the industry-standard tools he developed to count and analyze SLOC, he determined that it would cost over $1.2 billion to develop a Linux distribution by conventional proprietary means in the U.S. The Linux Foundation updated his study in 2008[2] and found it would take $10.8 billion to develop the Linux distribution Fedora 9 by traditional software development means in 2008 dollars.

We felt it would be interesting to use a similar approach and tools to analyze in simple terms the code present in each of the Linux Foundation's collaborative projects as of August 2015, and the required effort to re-create the R&D available to all. Of course, collaboration is happening throughout the industry in places like Mozilla, the Apache Software Foundation, GitHub, and so on. While this is a good starting point since the Linux Foundation today hosts many large-scale open source projects, it is by no means the full story.

[1] http://www.dwheeler.com/sloc/redhat71-v1/redhat71sloc.html
[2] http://www.linuxfoundation.org/sites/main/files/publications/estimatinglinux.html

We set out to give an estimate of the value of the code present in the projects by analyzing the code repositories of each one of our projects using the well known Constructive Cost Model (COCOMO) to estimate the total effort required to create these projects.

In short, the results are impressive.

As of last month, 115,013,302 total lines of source code were present in The Linux Foundation's collaborative projects. Using the COCOMO model, we estimated the total amount of effort required to retrace the steps of collaborative development to be 41,192.25 person-years. In other words, it would take a team of 1,356 developers 30 years to recreate the code base present in The Linux Foundation's current collaborative projects listed above. We estimate the total economic value of this work to be over $5 billion.

Let's discuss how we did the analysis.

The input for COCOMO is "KLOC," or thousands of lines of source code in the final project. For the snapshot comparison, we calculated the total lines of source code from each Linux Foundation Collaborative Project as it exists today, under the assumption that each would be developed in parallel. By using the COCOMO model, we were able to estimate the total value of all our collaborative projects. Please note this does not include the Linux kernel itself.

For this count, each project's source code was analyzed by David Wheeler's SLOCcount utility. We modified the latest version of SLOCcount, allowing it to count new programming languages like Go, which were not present when David Wheeler originally created the tool. All code, including our local modifications, is available here: github.com/licquia/sloccount

The interesting problem, in some cases, was determining "each project's source code." Some of the Linux Foundation's collaborative projects are quite complex, and often include code from outside projects. We strove to identify code developed as a part of the project itself, and not count these outside projects pulled in by reference. But we will freely admit this is not an exact science.

To do this, we used the following rules:

- We used each project's version control repository as the main source, rather than distributed source tarballs. (As it turns out, all projects included in this survey used Git.)
- When available, we also used lists of additional contributing projects as described by projects on their websites, or as pulled from each project's own documentation on how to contribute.
- In some cases, we used all of the repositories available in Git, with manual review to eliminate obvious upstream projects, such as Linux kernel trees.

Detailed Results

Given all the assumptions shown previously, the SLOC and estimated production values for all Linux Foundation collaborative projects as of August 2015 are as follows:

Total Physical Source Lines of Code (SLOC)	115,013,302
Development Effort Estimate, Person-Years (Person-Months) (Basic COCOMO model, Person-Months = 2.4* (KSLOC**1.05))	41,192.25 (494,306.95)
Development team size (Basic COCOMO model, Team Size = Person-Months / Months)	1356
Schedule Estimate, years (months) Basic COCOMO model, Months = 2.5* (person-months**0.38)	30.37 (364.47)
Total Estimated Cost to Develop (average salary = $95,280.00/year, overhead dividend = 0.693).	$5,663,488,007.63

Broken out by project we find:

PROJECT	LINES
Code Aurora Forum	23,137,096
OPNFV	787,665
Node.js	2,601,172
Tizen	80,016,660
OpenDaylight	2,231,469
Dronecode Project	1,733,498
OpenBEL	698,570
Xen	647,302
AllSeen Alliance	466,375
Cloud Foundry Foundation	1,261,690
OpenMAMA	306,906
IoTivity	300,187
Yocto Project	138,381
Cloud Native Computing Foundation	542,027
Let's Encrypt	89,574
AGL	18,839
Open Container Initiative	25,660
Total lines of code[3]	**126,922,318**

[3] Lines of code analyzed August 2015

The labor cost inputs for both salary and fully loaded costs were taken from the US Bureau of Labor Statistics. Wheeler's 2002 study and the Linux Foundation's 2008 update both used 2.4 for the fully-loaded overhead cost multiplier. The 2.4 multiplier came from informal discussions Wheeler had with several cost analysts[4]. Since we couldn't replicate these discussions, we decided to use the more conservative 0.693 overhead multiplier obtained from the US Bureau of Labor Statistics[5]. As a result this study contains more conservative estimates than previous studies, and straight up comparisons will be difficult to do.

Median software developer salary (year)[6]	$95,280.00
Fully-loaded cost dividend[7]	0.693
Fully-loaded developer cost	$137,489.1

The COCOMO model also takes project scaling issues into account when estimating developer time and cost, rather than assuming that developers scale in a linear fashion. This makes it possible to estimate the size of the developer team and the total time required using the model. Since the model gives us person-months of effort and months of time, dividing the first figure by the second yields the team size predicted by the model: 1,356 developers.

Limitations to this Study's Approach

As we said during our earlier analysis of Linux, there is no perfect way to estimate the value of something as complex and evolving as open source projects. While this method is one of the only viable approaches, it likely over-estimates some aspects of value, while under-estimating others. A few of the limitations in this approach:

Value Doesn't Really Equal Code.

Unfortunately this method equates value to quantity of code. We're only estimating what it would cost to recreate the codebases. It would be folly to look at the results and ascribe a complete value of a project through this method (even though to be fair that is what we have done!) Our projects span a huge swath of technology. Some have large amounts of code that started with the project. Some are lightweight but very powerful and crucial to the future of computing infrastructure. Some of the most valuable code derives its value in being small and highly efficient. Some have large blocks of code that are rarely used but still present in the project. It's impossible to tell by looking at the list of code in these projects which projects are "better" or "more important" purely based on lines of code. We discuss the ecosystem

[4] http://www.dwheeler.com/sloccount/sloccount.html

[5] http://www.bls.gov/news.release/archives/ecec_06102015.pdf

[6] http://www.bls.gov/oes/current/oes_nat.htm from May 2014

[7] http://www.bls.gov/news.release/archives/ecec_06102015.pdf

potential and adoption of some of these products later which is certainly another way to value code. In short, likely the best way to value code is just to see how it's used and what problems it solves.

Differences in Project Scope

Readers may also note that both Tizen and Code Aurora Forum have by far the most code of all the projects. This is not at all surprising given the nature of these collaborative projects. Tizen is a full mobile/embedded OS, much like a Linux distribution. And Code Aurora Forum recreates many full projects for a specific architecture. The data makes clear that there is a lot of development output in both projects, but apart from that a relative value compared to other projects should not be assumed.

Net Additions

The biggest weakness in SLOC analysis is its focus on net additions to software projects. Anyone who is familiar with kernel development, for instance, realizes that the highest labor cost in its development is when code is deleted and modified. The amount of effort that goes into deleting and changing code, not just adding to it, is not reflected in the values associated with this estimate. Because in a collaborative development model, code is developed and then changed and deleted, the true value is far greater than the existing code base. Just think about the process: When a few lines of code are added to an open source project, for instance, many more have to be modified to be compatible with that change. The work that goes into understanding the dependencies and outcomes and then changing that code is not well represented in this study. For a good discussion of this process, see the Linux Foundation's publications on who is developing Linux[8].

Not Capturing Debate

Collaborative development means you'll often have multiple individuals or groups working on different approaches to solving the same technical problem with only one of those approaches, or a hybrid of the approaches "winning" inclusion in the final version. Often much of the value is in the discussion and debate supporting different approaches. However, since the "losing" approaches are not committed to the final shipping release, this SLOC approach does not take into account the development effort for those approaches.

[8] http://www.linuxfoundation.org/publications/linux-foundation/who-writes-linux-2015

Starting from Scratch

This study assumes it would take the lines of code to recreate these projects if started from scratch. Certainly that is a specious assumption since there are probably many approaches to solving these problems.

Global

This study assumes all development would take place in the US, with the associated cost of US labor. Most software development is global in nature and its labor costs would vary accordingly.

Collaboration Builds Value Through Open Ecosystems

As mentioned above, while counting lines of code is at least one way to look at the value of a project, it's admittedly quite flawed. It does show effort of development and what it may take to recreate the code within these projects (even though of course that vary widely). But as we mentioned, the tasks and purposes of the projects are quite varied. A drone autopilot system is much different than a software defined networking controller. The true impact of a Collaborative Project is best seen by viewing where the code is used, the ecosystem impact of that code and the problem that code is solving.

For instance, because the Linux kernel is open, as more people use it in their products or services and those improvements or changes funneled back, the code itself becomes more valuable and improved. As Android became the most popular operating system in the world, huge value was built into the Linux kernel ecosystem by Android vendors. From hardware and board support to power management improvements, all of those changes were funneled back into the open source project and could be used by all. Thus Linux became a clear choice for other embedded or consumer electronics usage because the greater hardware ecosystem was already supporting it.

It also benefitted those who were not expecting it. Around the same time datacenter power, efficiency and space become a major concern, many of the same technologies that helped the mobile phone industry suddenly benefitted the enterprise. Battery saving on a phone also helped huge server farms cut down on their electric bills, helping us all. Those vendors' improvements and support are then also added back into the project: creating more jobs for engineers, better code and long term commitment from vendors who are shipping products for the code. This is why users like Facebook, Twitter, Google and so on have started open source projects of their own. It's a force multiplier for the quality and robustness of their code.

We are just starting to see these network effects in our collaborative projects as products, services, and infrastructure are dependent on the open code base. This is the true accelerant of value. While we have yet to come up with a reliable and objective way to quantify the ecosystem accelerant, the value is clear. The health of a code base can be determined by the number of developers, the lines of code, the rate of change and certainly market adoption. We also see strong projects develop interworking relationships with communities from other open source projects. Over time developers on other projects also contribute back as the ecosystem of dependencies expands.

Let's look at some examples of market adoption from Linux Foundation Collaborative Projects:

- Toyota and Jaguar Land Rover have publicly stated they will be using Automotive Grade Linux (AGL) in future production vehicles. Automotive Tier One suppliers such as Denso, Panasonic, and Fujitsu Ten have said they plan to base their future products on AGL. With a standardized app framework, AGL plans to create a robust application developer ecosystem. One of the largest automotive manufacturers already expects to ship 80 percent of its cars running AGL by 2016.

- Cloud Foundry is today used in a great number of products from large technology organizations. Products include: IBM BlueMix, Pivotal Cloud Foundry and Pivotal Web Services, HP Helion, SAP Hana Cloud Platform, GE Predix, CenturyLink AppFog, Verizon Enterprise Cloud, Huawei Web Services, Accenture's Cloud Platform and many more.
  - In just one of many examples, Allstate Insurance has publicly discussed how Cloud Foundry has taken developer startup time from months to minutes[9].
  - The Cloud Foundry platform is the core of GE's Predix Industrial Internet systems[10].

- It's estimated that 700 to 1,000 companies develop products or services on the Dronecode technology stack, which doesn't include the thousands of individual makers and members of DIY Drones who build products based on the code. Product examples include:
  - 3DRobotics' Solo UAV
  - Walkera recently introduced the QR X350 Premium copter, based on Dronecode:APM
  - Parrot's popular Bebop quadcopter, using a port of Dronecode's APM flight code

- More than 85 million devices currently run the AllSeen Alliance's AllJoyn framework. AllJoyn is included in every version of Microsoft Windows 10, every LG TV that ships with webOS, Hitachi Smart Wi-Fi Speakers, Monster SoundStage, Panasonic's wireless speaker system, and many more. Manufacturers representing more that half of all of the white-label goods produced in the world are collaborating to create an AllJoyn profile for their products, bringing in the era of the connected appliance to brands including LG, Haier, Insteon and Electrolux.

[9] http://blog.pivotal.io/pivotal-cloud-foundry/case-studies-2/allstates-andy-zitney-is-disrupting-how-insurance-does-technology

[10] GE's Predix demonstrates the CF platform as the core of Industrial Internet systems

- Node.js is used in many, many ecommerce, media and mobile sites, including Paypal, Fidelity, Netflix, the New York Times, Medium, LinkedIn, Yahoo, CBS, Pinterest, Ebay, GoDaddy, Uber, Zendesk and many more.
  - There are 2 million unique IP addresses installing Node packages per month, and 2 billion package downloads per month.
- More than 30 vendors, suppliers and consultancies are using key OpenDaylight components and architecture to build products, solutions, apps and services[11].
- The Open Daylight project's user base has grown significantly in 2015, largely in the telco space but also in the enterprise:
  - Tencent is reenvisioning its DCI network with SDN and ODL.
  - AT&T is deploying ODL for its global SDN controller.
  - Comcast is using ODL for network automation to start and is collaborating with others in industry on a key ODL project.
  - Telstra is using ODL in their WAN for network services.
  - Caltech Large Hadron Collider team is researching the use of ODL to distribute 200+ TB of data to 473 facilities around the globe; they consider ODL the de facto standard.
  - Telefonica is using ODL as its central controller and as a way to advance Net-IDE
- Many of the world's largest cloud providers run on Xen: AWS, Rackspace, IBM/Softlayer, Verizon cloud and Alibaba's Aliyun.

As you can see, even though it's very hard to ascribe a number to it, the value of the ecosystem potential of these projects is vast. From shared plumbing standards that benefit vendors through interoperability to unplanned innovation as an idea seeded for one use case that benefits another, we expect to see greatly accelerated value from these projects as we enter the next phase of commercial adoption. We look forward to seeing growth not only in the code bases of these projects over time, but in the sheer value of the codeas quantified by market impact for the participants who use it.

But don't mistake commercial adoption as commercial control. The true virtuous circle of open source software is that anyone can fully use and participate in these projects. Every one of the Linux Foundation's collaborative projects are open to any technical participant or user. Just as Linux' open participation and usage have given advanced technical assets to many Davids battling Goliaths, we anticipate these open projects will also provide a leveling effect to people throughout the globe.

[11] https://www.opendaylight.org/solutions-provider-directory

Conclusion

There is massive economic value being generated by large scale collaborative development in open source. Companies and individuals have literally billions of dollars of research and development available to them through these projects and the thousands of developers contributing to them. It's unmistakable: We can build software more quickly and cheaply by fully participating and supporting neutral collaborative development projects.

The market forces at work – where the code is used and supported by a greater ecosystem of participants – is truly where value is accelerated. It's difficult for a sole participant to compete against an open and thriving community: There are too many diverse actors with too much motivation. And it's always difficult to compete against free. This is why we see literally all major technology vendors adopting open source strategically.

So what does this study "prove?" That is always a subjective answer, and we are the first to admit the limitations to these types of constructed analysis. But we do think it's clear that the complexity present in modern day software requires an economic investment that is unlikely to be shouldered by one company alone. From cloud computing to new ways of developing and deploying applications, the future of computing is open and collaborative. And for that we -- and users -- are thankful.

The Linux Foundation has identified that its work in delivering best practices for hosted collaboration has an acceleration effect on the value of these projects. By providing the "architecture of participation" via best practices and services gleaned from our years supporting Linux, The Linux Foundation is helping companies and individuals collaborate and innovate with speed and scale never before seen. We are hopeful that the many issues facing our planet can be solved by more large-scale open source collaborative development[12]. We invite interested projects, companies and individuals to join our existing projects or start one of their own and add to the $5 billion dollars of value already created in just the last few years.

More information can be found at collabprojects.linuxfoundation.org.

Acknowledgements

The authors would like to thank Dawn Foster, Brian Warner, Mike Dolan and David Wheeler for their contributions to this paper.

[12] https://www.youtube.com/watch?v=SqXUu-EsAiE

The Linux Foundation promotes, protects and standardizes Linux by providing unified resources and services needed for open source to successfully compete with closed platforms.

To learn more about The Linux Foundation or our other initiatives please visit us at www.linuxfoundation.org