An Overview of Open-Source Software Licenses and the Value of Open-Source Software to Public Health Initiatives

Erin N. Hahn

The use of open-source software (OSS) has dramatically increased in the past several years, particularly in the public health domain. The Johns Hopkins University Applied Physics Laboratory's (APL) work on developing and licensing OSS identified a need within the public health community to better understand the definition and connotations of the words open source *and the various open-source licenses. The use of OSS in the public health domain can dramatically improve the implementation of mobile and electronic health initiatives in resource-limited settings because OSS provides an affordable alternative to costly proprietary software.*

INTRODUCTION

The term *open source* generally refers to software that is made readily available by an individual or group for others to use, modify, or redistribute under a licensing agreement with very few restrictions. Anyone can use the software without having to pay royalties or negotiate a license agreement. Open-source software (OSS) is not a new creation, but it has been used with increasing popularity in large-scale commercial software projects in recent years. It has been called "the software that runs the Internet," referring to its significant use in the Internet's infrastructure, including the Apache Web server, the Mozilla browser, and Linux operating system.[1] There are currently at least 50 different open-source licenses, and they represent a unique approach to licensing when compared with licenses normally used in a commercial environment. (See Table 1 for a listing of licenses by name.[2])

In the past several years, the use of OSS in the public health field has grown dramatically. In particular, the field of mobile health, or mHealth, has seen a substantial increase in the use of OSS due to the ubiquitous nature of cellular telephones.[3] By definition, mHealth refers to "the practice of medicine and public health, supported by mobile devices."[4] In general, it involves the use of "mobile communication devices, such as mobile phones and PDAs, for health services and information."[5] Many of these platforms and associated tools, whose primary users are those in the field of public health or clinical care, are purportedly open source. However, in a community that is generally not as savvy in information technology, the term *open source* is confusing and leaves many unanswered questions. The Johns Hopkins University Applied Physics Laboratory (APL), in conjunction with the Global Emerging Infections Surveillance and Response System,

a division of the U.S. Armed Forces Health Surveillance Center (AFHSC-GEIS), has been working to develop and deploy open-source disease surveillance capabilities in resource-limited settings. In the course of this work, APL and AFHSC-GEIS identified a need to better understand the definition, nuances, and connotations of the term *open source* within the public health community.

The purpose of this article is to provide an introduction to open-source licensing and the main elements to consider when determining whether to use OSS or when selecting an open-source license. In particular, it provides examples of the use of open-source licensing in the development tools used in the public health domain. This article also provides background on copyright and the distinction between a copyright and a license, a discussion of the history of open source and "free software" (terms that are often used interchangeably and also frequently consolidated and referred to as FOSS, or Free and Open-Source Software), and an overview of commonly used licenses with strong user communities. It also discusses several myths related to the benefits and hazards of using open-source licenses and OSS and provides examples of how government agencies are confronting these myths and successfully using OSS to their advantage.

BACKGROUND ON COPYRIGHT

Copyright is a form of intellectual property law and protects original works of authorship.[5,6] All software is subject to copyright law, and as soon as source code is created, anyone (other the author) who wants to use it must obtain explicit permission from the author. (As soon as a work is created and "fixed in a tangible form that is perceptible either directly or with the aid of a machine or device," it is protected by copyright.[1,6,7]) Copyright is how an author retains control over his or her work. Software copyright is "the exclusive legal right to control the rules for copying, modifying, and distributing a work of software."[7] The person or organization that has the right to control the work is called the copyright holder. When copyright holders permit others to use, modify, or distribute their software, they have granted a license.[7] The license is the permission to use the software in some way—it can be an unconditional grant of permission that mirrors the rights of the copyright holder or a conditional grant of permission that allows individuals to copy or use the software according to certain provisions. Open-source licenses fall into both categories.

A general commercial copyright license usually protects the copyright holder's interests by placing restrictions on how the software can be used. For example, a commercial software license usually prohibits the copying or modification of the software, mainly by distributing only the machine-readable binary or object code. OSS licenses give users more rights than a general commercial license because the user gets access to the source code and has the right to change the source code. The term *copyleft* was generated by the free software community and is the term for a license condition that ensures that all modified versions of the software can be copied, modified, and/or distributed in the same way as the original. By ensuring that downstream users receive source code and permission to modify it, a copyleft license is said to keep code "forever free."[7] Not all open-source licenses have copyleft provisions, but many do. Despite what the name may imply, copyleft is still a license and is enforced by copyright law. Instead of withholding permission to copy or modify a work (as in the traditional sense of a copyright), copyleft uses copyright law to actually require that those permissions be granted.

It is important to note that using an open-source license is not the same as placing the software in the public domain. The terms of the open source license must be met, and the copyright holder retains rights to the work. If the terms of the open-source license are not met (e.g., the same licensing provisions are not applied to derivative works), the copyright has been infringed and the copyright holder has certain legal remedies available to him or her.[5] No matter how permissive the open-source license, the copyright holder's interests are protected. If the software is released into the public domain, the author surrenders the copyright. Put another way, as soon as software code is created and saved, a copyright attaches to the work. Placing the software in the public domain relinquishes all rights associated with the work. It is not equivalent to granting a license because the author is not limiting or placing restrictions on the use of the software in any way. In fact, software placed in the public domain can be used, modified, and removed from the public domain by another user asserting copyright ownership.[5]

FREE VERSUS OPEN SOFTWARE

The terms *free software* and *open software* are often used interchangeably, and they are also frequently consolidated and referred to as FOSS. However, although a majority of software is both open and free, there are distinctions to be made between the two categories, both in philosophy and in the licenses that fall within each. The free software movement is, at its roots, about the users' freedoms, whereas open source focuses on making software better from a practical perspective by allowing access to the code so that others to improve it. These two views may lead to the same outcome in how the software is treated from a copyright perspective, but the goals for getting to that outcome are slightly different. (For a comprehensive overview of free software and OSS, see Ref. 8.)

The concept of free software began in 1984 with Richard Stallman, a Massachusetts Institute of Technology (MIT) researcher. Stallman was concerned that computing would be dominated by a few powerful people

if software were all proprietary. He considered software scientific knowledge that should be shared and distributed to further innovation in computer science. In 1984, he left MIT and began the GNU Project and the Free Software Foundation (GNU is a recursive acronym for "GNU's Not Unix"). One goal of the GNU Project was the development of a freely available operating system that could run GNU software.[8]

The most important characteristic of free software is the underlying philosophy for why software should be free and what free means. The philosophy of free software is one that respects users' freedoms while benefiting society by promoting sharing and cooperation. The term *free software* is about freedom, not price. The distinction Stallman makes is that free software is "free as in speech, not as in beer."[8] A program is free if you can run the program for any purpose; you have the freedom to modify the program (requiring access to the source code); you have the freedom to redistribute copies with or without a fee; and you have the freedom to distribute modified versions of the program so the community of software developers can benefit from improvements.[8,9] Note that the freedom to sell copies of software is permissible because the term *free* here does not refer to price, so there is nothing prohibiting someone from generating revenue from free software—the founders of the free software movement believed such revenue could ideally be used to generate new free software projects. However, to thwart businesses from co-opting free software for their exclusive commercial use, Stallman created the GNU General Public License (GPL). The GPL license is discussed in detail later in this article.

Many software companies rejected the concept of free software in part because it seemed to fundamentally conflict with having and furthering a commercial interest in a product. So, in 1997, a group of individuals came together to promote the concept of free software and created the term *open source*.[8] Using the term *open source* was a way to market the idea of free software by removing the economic context of "free" to make it more palatable to private companies. However, there are practical differences between free software and OSS. While open source captures much of the spirit of GNU, it allows for provisions free software does not, such as the ability to mix proprietary software and OSS. The Open Source Initiative (OSI), an organization that provides oversight of the open-source mission, refers to open source as "a development method for software that harnesses the power of distributed peer review and transparency of process. The promise of open source is better quality, higher reliability, more flexibility, lower cost, and an end to predatory vendor lock-in."[10]

The OSI created the Open Source Definition, which has several distribution terms with which OSS must comply.[10] According to OSI, software is open source if it meets the following criteria:[11]

"1. **Free Redistribution.** The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2. **Source Code.** The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

3. **Derived Works.** The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. **Integrity of the Author's Source Code.** The license may restrict source code from being distributed in modified form *only* if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5. **No Discrimination Against Persons or Groups.** The license must not discriminate against any person or group of persons.

6. **No Discrimination Against Fields of Endeavor.** The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. **Distribution of License.** The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. **License Must Not Be Specific to a Product.** The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9. **License Must Not Restrict Other Software.** The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open source software.
10. **License Must Be Technology Neutral.** No provision of the license may be predicated on any individual technology or style of interface."

OVERVIEW OF COMMONLY USED LICENCES

In general, open-source licenses can be broadly categorized into those that apply no restrictions on the distribution of derivative works and those that do apply restrictions to ensure that the code will always remain open/free. (Note that the term *derivative works* is defined by the U.S. Copyright Act and generally refers to a work based on one or more preexisting works. However, the Act does not specifically address derivative works in software, so the law as it applies to OSS is not well established.) The former is also called an "academic license," and its purpose is to promote a public commons with unlimited use but it contains no requirement to contribute back to the community.[12] The latter type of license is also referred to as a reciprocal or "share alike" license because it requires that any derivative work retain the original license. Although there are licenses that exist outside of these categories, the licenses discussed below are grouped into one of these two categories, with the exception of the Mozilla license, which is characterized as a hybridization of both. Table 1 summarizes the most commonly used open-source licenses, although additional licenses are included in the discussion below.

Academic or Nonprotective Licenses

The Berkeley Systems Distribution License

The Berkeley Systems Distribution (BSD) license is one of the least restrictive and most recognized open-source licenses.[13–15] The license was developed by the University of California at Berkeley and allows free use of the OSS, including the ability to modify the software. The BSD license allows for redistribution and use of source code whether modified or not, as long as the source code retains the copyright notice and other notices regarding disclaimers of warranty and limitations on liability found in the license.[13] The BSD license allows the software to be combined with proprietary software or modified and turned into proprietary software. The BSD allows derivative works to be released under a license other than the BSD; hence there is no copyleft provision. The original BSD had a clause mandating attribution of contributors in advertising of the software. This clause has since been removed, although users of code under the old version of the BSD license must be careful to comply with the advertising clause. A clause still exists prohibiting use of the copyright holder's name in any promotion of software.

The MIT License

The MIT license is very similar to the BSD license and is often referred to as being part of the BSD family of licenses. Like the BSD license, it permits reuse of open-source code within proprietary software as long the MIT licensing terms are included in the proprietary software. The main differences between the MIT license and the BSD license are that the MIT license does not contain a clause prohibiting the use of the copyright holder's name in promotion of the software and it places more emphasis on the user by emphasizing the right to "use, copy, modify, merge, publish, distribute, sublicense and/or sell copies of the Software."[16]

Apache

The Apache license was created by the Apache Software Foundation. Like the BSD and MIT licenses, the Apache license allows software to be used without any obligation to redistribute the source code of any of the derivative works. The main difference is that the Apache license provides a clause about patent licensing and termination. In addition to providing a patent clause, the Apache license requires that any modifications to the source code distributed under the license carry prominent notices that the files were changed.

Artistic

The Artistic license falls into the same category as the BSD, MIT, and Apache licenses in that it allows modified versions of Artistic software to be licensed independently, with some conditions.[17] Version 1.0 was criticized by the Free Software Foundation for being too vague, but version 2.0 is accepted by both the Free Software Foundation and OSI. The Artistic license was the first open-source license deemed enforceable under copyright law as opposed to contract law.[18]

Mozilla

The Mozilla Public License (MPL) was originally created by Netscape, and version 1.1 is used by the Mozilla Application Suite, Mozilla Firefox, and other Mozilla software and has been adapted for use by other companies.[15] It combines aspects of the BSD and GPL licenses. It allows for commercial licensing of derivative works, and changes to source code covered under the license must be made freely available. Additions to source code that are not modifications and contribute to a larger work can be licensed under something other than MPL

Table 1. Licenses used by a sample of open-source initiatives

Open-Source Initiative	Description	License Used
Frontline SMS	Designed for grassroots nongovernmental organizations in developing countries, it helps organizations overcome communication barriers by allowing users to send, receive, and manage short message service (SMS) over a mobile network.	LGPL
Java Rosa	Open-source platform for data collection on mobile devices	Apache 2.0
Rapid SMS	Open-source framework for dynamic data collection, logistics coordination, and communication leveraging basic SMS mobile phone technology	BSD
SAGES (Suite for Automated Global Electronic bioSurveillance)	Collection of modular, flexible, OSS tools for electronic disease surveillance	Apache 2.0
RapidAndroid	A fully featured implementation of Rapid SMS that uses the mobile device to act as a standalone appliance for SMS management	Apache 2.0
Ushahidi	Initiative that creates OSS for information collection, visualization, and interactive mapping	LGPL
ODK	Free and open-source set of tools that help organizations author, field, and manage mobile data collection solutions	Apache 2.0
OpenXData	Open-source data-collection platform that supports low-cost mobile phones	Apache 2.0

and do not have to be published. In this sense, the MPL is not a strong copyleft license like GPL because the rights contained in the license must be preserved only for modifications to MPL source code but not for additions extending the software. Thus, the code can be used to create a proprietary product, and the additions to the code can be licensed in a closed source manner.[19,20]

Reciprocal or Protective Licenses

GNU GPL

The GNU GPL was written by Richard Stallman for the GNU Project and is the most widely used free software license. The license has been described as a manifesto because the license itself contains language about the freedom of software and the rationale behind the creation of the license.[15] It is the first copyleft license. As mentioned earlier, copyleft describes the requirement that all derived works must be distributed under the same license. Therefore, it does not allow users to modify GPL programs and make them private or proprietary.

The license takes a unique approach at guaranteeing freedom in that it uses restrictions in the license to protect users' rights to freely use software, as opposed to outlining what is prohibited or how the license prohibits use. For example, it states:

> When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.[21]

The GPL does not allow software licensed under a GPL to be combined with a proprietary program, because a proprietary program will not give a user as many rights as the GPL (fundamental to the notion of copyleft).[21] Any software using a GPL must, when distributed, publish the copyright notice and disclaimer of warranty on each copy and provide recipients of the program with a copy of the license. By modifying or distributing GPL software, a user is deemed to have accepted the terms of the GPL.

Lesser GPL

The copyleft nature of the GPL and the concern that any code written in GPL incorporated into another program will require the second program to be licensed under GPL, no matter how small a portion of the code is originally GPL, led to the development of another license.[15] The Lesser GPL (LGPL) was created to allow proprietary software to be used with GPL software through the use of programming libraries (which is why it is sometimes referred to as the Library GPL). The LGPL allows the proprietary software incorporated with GPL-licensed software through a library to be licensed independently from the GPL. It is a compromise between the strong copyleft nature of the GPL and more permissive licenses such as the BSD. The LGPL allows GPL software to be linked to a non-GPL program regardless of whether it is free. In the case of programming libraries, the GPL software can be used by the library (and hence linked to other programs).[22] Despite the fact that the Free Software Foundation created the LGPL, it does not encourage its use, mainly because with the exception of

limited circumstances, it does not further the interests of free software developers.[23]

Eclipse Public License

The Eclipse Public License (EPL) replaced a license called the Common Public License. It has weaker copyleft provisions than the Common Public License had, and it also has a patent clause. Additions to source code originally published under an EPL license can be licensed in another way as long as the additions do not constitute derivative works of the EPL-covered source code but act as "separate modules" of software.[24] Derivative works under EPL must also be licensed as EPL, which makes it a limited copyleft, a characteristic of the GPL license. However, the EPL requires that anyone distributing the work grant all recipients rights to any patents that may cover modifications. This patent clause is a restriction that is not compatible with GPL, so EPL and GPL works cannot be combined and legally distributed, but combined works using other licenses are permissible.

COMMON MISCONCEPTIONS ABOUT OPEN-SOURCE LICENSES

Open Source Means Free

This particular misunderstanding is perhaps the most common and is linked to the "free as in beer" way of thinking about free. OSS is provided to users at no cost, but this does not mean implementing OSS is free of cost. Although the software costs no money to download, which makes it accessible to a broader community of users, the assumption that there is no cost of ownership is faulty. For example, installation and integration of the software often requires technical expertise, and this cost can strain development budgets if the integration is complex. Maintenance of the code is another cost and requires the time of either in-house or external developers. Moreover, the defining characteristic of open source is really the ability to access the source code and not as much the fact that the source code is made available at no cost.

Open Source Has No Copyright Restrictions

As discussed previously, but worth emphasizing again, providing software as open source does not mean the developer has relinquished copyright protection. In fact, how a user is able to exploit the source code and restrictions on that use varies by license, each of which protects the rights and intent of the original author. Open-source licenses are grounded in copyright law, and the copyright holder gets to choose which rights are granted to other users.

OSS Is Unreliable

There are two components to this myth. The first part is asserted on the basis that the software is unreliable because it is either produced by amateurs or circulated without being tracked for bugs or quality. The second part of the myth is that the software is unreliable in the sense that it is uniquely insecure because vulnerabilities in the code can be easily detected. As for the first assertion, and as discussed in the introduction of this article, large, commercial software projects use OSS, and there is a high level of demand for the use of OSS in many domains. Although the software is not necessarily tracked for quality, and the licenses may not assert warranties for fitness, some software projects do have managers tracking code. Moreover, providing accessibility to a broad group of users is one of the reasons software is made open source—so others can improve the existing code. This logic speaks to the security issue as well. The transparency of OSS and the ability to improve the software are reasons many consider it more secure.

ENFORCEABILITY

The legal enforceability of open-source licenses is a nuanced and developing area. The cases vary depending on the license at issue and the facts around which enforcement is sought. Given the philosophical underpinnings of OSS, it was not immediately clear whether certain contract elements, namely the exchange of consideration, could be met given the free nature of the license. There is a growing body of case law, but for purposes of this article, it is primarily important to note that open-source licenses are enforceable under both contract and copyright law.

The case of *Jacobsen v. Katzer*[18] highlights the enforceability through both legal mechanisms. In *Jacobsen*, the court found that if a licensee breaches a condition placed on the license grant, the licensor's copyright has been infringed.[18] The court also found that injunctive relief can be granted for open-source licenses.[18] Injunctive relief is relief granted by a court against an act or condition, as opposed to a grant of money damages. An example in this context may be an order to stop distribution of the software by those not complying with the license terms. This type of relief is particularly important in the open-source community because monetary damages, which are typically sought in contract cases, may not be an available option as the software may have been distributed without profit. *Jacobsen* also confirmed that open-source licenses do not lack consideration and can therefore be enforced under contract law.[18]

BENEFITS AND LIMITATIONS OF OSS

The open-source model has been very successful and provides developers with many benefits. First, access to

source code enables developers to improve the code, create programs that are more interoperable, and perfect their own programs that they are using OSS to develop. Access allows others to build on software in ways not envisioned by the original creators. This ability to access source code is, in part, due to the strong communities around many types of open-source licenses, and these communities provide a large pool of code from which to work. Open-source licensing provides developers who may not otherwise be able to pay for a program access to the source code, as most programs distributed under open-source licenses are free. This is particularly important in the case of resource-limited countries, which need access to similar software but do not have the means to pay for or sustain a proprietary software license. Most importantly, the broad rights that are granted to users through an open-source license provide a significant benefit because they allow users to modify, use, or distribute the software, whereas commercial licenses are usually distributed only in a form that cannot be modified.

Despite the many benefits of OSS, users of this software and those selecting an open-source license must be aware of the distinctions between the types of licenses, no matter how seemingly trivial the distinctions can be. A major consideration is whether the user wants derivative works to be proprietary, in which case a copyleft license would not be appropriate. Although there are various specialized licenses to address unique circumstances, if the developer wants to make the program open source to tap into the development community, he or she will want to pick a license that is easy for other developers to work under—probably a standard and widely used license.

Although the accessibility of the software is a fundamental characteristic of open source, most licenses contain disclaimers concerning warranties and fitness for a particular purpose. Although there is no definitive evidence suggesting that OSS is of lesser quality than commercial software (as indicated above, some in fact argue the opposite), the licensee may have to accept risks that the software has major errors. Some initiatives are large enough to provide code monitoring and bug tracking, but this is not always the case. Also, the fact that numerous people are contributing to the code increases the likelihood that code infringing on intellectual property rights (here, perhaps certain copyright terms) is introduced. Most licenses disclaim all warranties, and it may be difficult to audit the code to determine which contributor or contributors may have violated the terms of the license. An open-source project that has many authors, each of whom has a license on his or her work, makes determining who can enforce the copyright difficult (i.e., determining whether one owner can bring an action on behalf of all copyright owners or whether each must be found and joined in an action). However, the idea that OSS is more prone to claims of intellectual property infringement is generally not supported by fact, even though there have indeed been such claims against open-source development projects and there will likely continue to be such claims. The existence of these claims alone does not support the conclusion that OSS is especially vulnerable. It does, however, emphasize the enforceability of open-source licenses as legitimate intellectual property claims that can be brought before a court.

GOVERNMENT USE OF OSS

Until recently, government use of OSS has been limited, but with the expansion of mobile and cloud computing, more agencies are adopting policies for using OSS on government-funded projects.[25] Some of the government's reluctance stems from the sensitivity of certain data and concerns about information assurance. However, because OSS goes through continuous peer review, some argue that it is more secure than proprietary software.[25,26] In particular, the DoD and NASA have embraced the use of OSS, with the latter agency being referred to as "the summa cum laude when it comes to open source" since using OSS to develop cloud computing networks.[25] The Department of Homeland Security created the Homeland Open Security Technology (HOST) program to leverage the use of OSS in the development of technologies to support cybersecurity objectives.[27]

In a DoD-circulated memorandum, the DoD confronted many of the previously discussed misconceptions about OSS and stated that "there are many OSS programs in operational use by the Department today, in both classified and unclassified environments."[26] The memorandum specifically advises that as part of the market research federal agencies must conduct to procure property or services, OSS should be included in the research when it meets mission needs.[26] Moreover, it points out that many open-source licenses allow the user to modify the OSS for use with no obligation to redistribute, therefore quelling the misunderstanding that the DoD or any government agency would have to distribute the source code to the public, which would be prohibited on classified projects. (Note that the memorandum does outline conditions under which the code should be distributed to the public and essentially states that doing so must be in the government's interest; the government must be authorized to release the code; and public release cannot be otherwise restricted by law.[26]) The memorandum underscores other benefits of using OSS, including the following:[26]

- The ability to "respond more rapidly to changing situations, missions, and future threats" because of the unrestricted ability to modify source code
- The identification and elimination of defects through the "continuous and broad peer-review enabled by publicly available source code"

- The availability of the code for maintenance and repair by the government and its contractors (rebutting the notion that OSS comes with a limited or no warranty and therefore should not be used)
- The ability to reduce reliance on a particular vendor due to the use of OSS, which can be maintained by a variety of vendors
- The cost advantage provided by OSS, as it typically does not have a per-seat licensing cost
- The ability to widely disseminate the software, which allows the agency to contribute to a collaborative software development environment, particularly one run by the Defense Information Systems Agency (www.forge.mil/Community.html)

The work of the DoD, NASA, and the Department of Homeland Security will undoubtedly help set the trend for broader use of OSS by the government. However, the development of mobile technologies for the government is also stimulating increased use of OSS. Government-deployed mobile applications are an area of growth, and many agencies are interested in using mobile operating systems like Google's Linux-based Android for development.[23] mHealth initiatives and the need for electronic processes to support healthcare (eHealth) provide particularly good examples of government use of OSS.

USE OF OSS FOR PUBLIC HEALTH INITIATIVES

The growth of global and national mHealth and eHealth needs has spurred innovation in software development. As medical practitioners and health institutions are encouraged by the federal government to digitize patient information to reap efficiency and productivity benefits of digital information, a need for sophisticated tools has arisen.[28] In addition, in areas that are resource limited but where cellular technology is prevalent, mHealth solutions can dramatically improve the ability of local and nonprofit public health organizations to harness the power and potential usefulness of large amounts of health data. The monetary costs of licensing and maintaining proprietary software systems have been common challenges to these end users. Fortunately, the OSS paradigm has gained strong worldwide acceptance, and grassroots entities, researchers, and nonprofit institutions are on the frontier of developing innovative open-source tools to fulfill user needs.

The Suite for Automated Global Electronic bioSurveillance (SAGES) program at APL has been involved in the mHealth and eHealth open-source space since 2007 and has developed three open-source tools: ESSENCE Desktop Edition (EDE), OpenESSENCE, and a short message service (SMS) data collector. All three of these tools leverage OSS—this was a key design factor to ensure affordability and sustainability.

In addition to SAGES, other open-source tools have been developed by several groups to fit various needs for mobile-based data collection. Some tools and frameworks leverage others, whereas some were new software architectures entirely.

The wide selection of OSS licenses gives developers options to license their work depending on their preferences. Consequently, it is important for end users to evaluate licensing of a third-party tool before integrating it with their own projects to ensure that the license terms are not violated.

Besides mHealth and eHealth applications, the use of OSS has played a significant role in postdisaster areas such as Haiti and, most recently, the Philippines. OpenStreetMap (OSM) is a crowd-sourced mapping application that provided a detailed map of the areas hit by typhoon Haiyan within 3 days of landfall.[29] The Red Cross used OSM to coordinate the volunteer effort in the Philippines, and the organization now has a policy of using OSS in all of its projects. A key reason the Red Cross cited for the adoption of this policy is the reduction or elimination of sustainment costs of software after the organization leaves an area.

In the aftermath of the 2010 earthquake in Haiti, several crowdsourcing applications such as OSM were used to map the damage. Ushahidi (http://www.ushahidi.com/about-us), a company that develops OSS for information collection and interactive mapping, led one of the main volunteer efforts to produce a crisis map.[30] In an independent evaluation of the use of the Ushahidi Platform—Ushahidi's collection, visualization and interactive mapping tool—in Haiti, a report produced by a team of independent consultants noted that Ushahidi's mapping effort provided critical situational awareness that influenced operational and tactical decisions and saved lives.[31] The report urged that stronger support from the nongovernmental organization community would be useful in making the application more widely used in the response community, something OSM was able to do through its relationship with the Red Cross in the Philippines. It is worth mentioning that Ushahidi staff worked collaboratively with OSM and other applications as sources of information. This partnership was critical to the company's effectiveness. The ability to share information and provide near real-time updates was facilitated by the fact that all the technology was using OSS and relying on volunteer input to improve the information's accuracy.

CONCLUSION

The variety of open-source licenses and the user-community support that accompany many of them offer tangible benefits to governmental and nongovernmental entities wishing to use and develop OSS. With the increased use of OSS by the government, many of the

misconceptions, particularly regarding risks to the user, have been dispelled. Using OSS or licensing software with an open-source license requires a clear understanding of the distinctions between the licenses and the obligations that each require for use and sharing of such software.

The public health domain provides excellent examples of how using OSS can spawn collaboration and technological advances. OSS in the public health field is a rapidly evolving space that drives innovation and brings needed tools into the hands of those often disenfranchised because of a lack of financial assets. Although skeptics have cited decreased quality and instability of OSS, there are numerous examples of reliable OSS applications that have had a significant impact in resource-limited areas.

ACKNOWLEDGMENTS: I thank my APL colleague Adjoa Poku for sharing with me her knowledge of open-source tools for mobile-based data collection and for her feedback on the development of an earlier version of this article.

REFERENCES

[1]Kennedy, D., "Best Legal Practices for Open Source Software: Ten Tips for Managing Legal Risks for Businesses Using Open Source Software," *Law and Technology Resources for Legal Professionals*, http://www.llrx.com/features/opensource.htm (7 Feb 2006).
[2]Open Source Initiative, *Licenses by Name*, http://www.opensource.org/licenses/alphabetical (accessed 27 Mar 2014).
[3]Terry, K., "Mobile Health: New Cell Phone-Based Technologies Transform Emergency Care," *CBS Money Watch*, http://www.cbsnews.com/8301-505123_162-43841703/mobile-health-new-cell-phone-based-technologies-transform-emergency-care/ (last modified 1 Sep 2010).
[4]Wikipedia contributors, "mHealth," *Wikipedia, The Free Encyclopedia*, http://en.wikipedia.org/wiki/MHealth (accessed 27 Jan 2012).
[5]The Copyright Act, 17 U.S.C. § 101 *et seq.*
[6] U.S. Copyright Office, *Copyright in General*, http://www.copyright.gov/help/faq/faq-general.html#what (last modified 12 Jul 2006).
[7]Fontana, R., Kuhn, B. M., Moglen, E., Norwood, M., Ravicher, D. B., et al., "Common Copyright Questions," Chap. 2, *A Legal Issues Primer for Open Source and Free Software Products*, Version 1.5.2, Software Freedom Law Center, New York, http://www.softwarefreedom.org/resources/2008/foss-primer.html#x1-40002 (4 Jun 2008).
[8]DiBona, C., Ockman, S., and Stone, M., *Open Sources, Voices from the Open Source Revolution,* O'Reilly Media, Sebastopol, CA (1999).
[9]Free Software Foundation, *What is Free Software? The Free Software Definition*, http://www.gnu.org/philosophy/free-sw.html (last modified 8 Mar 2014).
[10]The Open Source Initiative, http://opensource.org/about (accessed 22 Jun 2014).
[11]Open Source Initiative, *The Open Source Definition*, http://www.opensource.org/docs/osd (accessed 13 Mar 2014).
[12]Rosen, L., "Choosing an Open Source License," Chap. 10, *Open Source Licensing: Software Freedom and Intellectual Property Law*, Prentice Hall, Upper Saddle River, NJ, p. 230 (2005).
[13]Open Source Initiative, *The BSD 3-Clause License*, http://www.opensource.org/licenses/BSD-3-Clause (accessed 13 Mar 2014).
[14]Goldstein, D. E., Ponkshe, S., and Maduro, R., *Analysis of Open Source Software (OSS) and EHRs: Profile in Increasing Use of OSS in the Federal Government and Healthcare*, Market Update Report, Medical Alliances, Inc. (2004).
[15]Kennedy, D. M., *A Primer on Open Source Licensing Legal Issues: Copyright, Copyleft, and Copyfuture*, National Technical Institute of Athens (Greece) (Glotta NTUA), http://www.cs.miami.edu/~burt/learning/Csc322.052/docs/opensourcedmk.pdf (accessed 22 Jun 2014).
[16]Open Source Initiative, *The MIT License*, http://www.opensource.org/licenses/MIT (accessed 13 Mar 2014).
[17]Open Source Initiative, *The Artistic License 2.0*, http://www.opensource.org/licenses/Artistic-2.0 (accessed 13 Mar 2014).
[18]*Jacobsen v. Katzer*, 535 F.3d 1373 (Fed. Cir. 2008).
[19] Mozilla, *MPL 2.0 FAQ*, http://www.mozilla.org/MPL/2.0/FAQ.html (accessed 13 Mar 2014).
[20]Open Source Initiative, *Open Source Initiative OSI - Mozilla Public License 1.1 (MPL-1.1): Licensing*, http://www.opensource.org/licenses/MPL-1.1 (accessed 13 Mar 2014).
[21]GNU Operating System, *GNU General Public License, Version 3*, http://www.gnu.org/licenses/gpl.txt (29 Jun 2007).
[22]GNU Operating System, *GNU Lesser General Public License, Version 3*, http://www.gnu.org/licenses/lgpl.txt (29 Jun 2007).
[23]GNU Operating System, *Why You Shouldn't Use the Lesser GPL for Your Next Library*, http://www.gnu.org/licenses/why-not-lgpl.html (updated 29 Jul 2013).
[24]Open Source Initiative, *Eclipse Public License 1.0 (EPL-1.0)*, http://www.opensource.org/licenses/EPL-1.0 (accessed 13 Mar 2014).
[25]Brodkin, J., "Nonprofit Helps Government Expand Open Source Software Usage," *NetworkWorld*, http://www.networkworld.com/news/2011/062711-government-open-source.html (27 Jun 2011).
[26]Department of Defense, *Clarifying Guidance Regarding Open Source Software (OSS)*, Memorandum, http://dodcio.defense.gov/Portals/0/Documents/FOSS/2009OSS.pdf (16 Oct 2009).
[27]U.S. Department of Homeland Security, *Cyber Security R&D Center*, http://www.dhs.gov/csd-host (accessed 22 Jun 2014).
[28]Centers for Medicare & Medicaid Services, *EHR Incentive Programs: The Official Web Site for the Medicare and Medicaid Electronic Health Records (EHR) Incentive Programs*, https://www.cms.gov/EHRIncentivePrograms/ (last modified 18 Feb 2014).
[29]Robinson, M., "How Online Mapmakers Are Helping the Red Cross Save Lives in the Philippines," *Atlantic*, http://www.theatlantic.com/technology/archive/2013/11/how-online-mapmakers-are-helping-the-red-cross-save-lives-in-the-philippines/281366/ (12 Nov 2013).
[30]Patrick Meier, "How Crisis Mapping Saved Lives in Haiti," *National Geographic NewsWatch*, http://newswatch.nationalgeographic.com/2012/07/02/crisis-mapping-haiti/ (2 Jul 2012).
[31]Morrow, N., Mock, N., Papendieck, A., and Kocmich, N., *Independent Evaluation of the Ushahidi Haiti Project*, http://ggs684.pbworks.com/w/file/fetch/60819963/1282.pdf (8 Apr 2011).

The Author

Erin N. Hahn is a Senior Professional Staff member in the National Security Analysis Department at APL. She conducts analysis on projects involving policy issues related to irregular warfare, international law, and information privacy. She also contributes to studies on insurgencies and unconventional warfare and supports the PRedicting Infectious disease Scalable Method (PRISM) project in APL's Asymmetric Operations Sector. She previously worked as an attorney in private practice. Her e-mail address is erin.hahn@jhuapl.edu.

The *Johns Hopkins APL Technical Digest* can be accessed electronically at **www.jhuapl.edu/techdigest**.