RESEARCH ARTICLE

Open Source and Public Health: Building on a Tradition of Collaboration

Authors
Noam H. Arzt, PhD, HLN Consulting, LLC
Michael Berry, HLN Consulting, LLC

Correspondence
Noam H. Arzt
Email: arzt@hln.com

Abstract

While there have been significant advances in open source software development and products available to public health, there appears to be much confusion around this topic related to open source licensing, management, fair use, and implied cost. This article will provide needed definitions of open source and proprietary software; describe the strengths, weaknesses, opportunities, and threats (SWOT) around each; and provide examples of key open source products in use outside of public health. It postulates that the shared tradition of collaboration in both public health agencies and open source communities provides a unique opportunity for the success of open source in public health; and explores this with a case study of open source management and community which may serve as an exemplar of how an open source perspective can fuel a healthy alternative to the proprietary software market in public health.

Keywords: open source, software, public health, EHR

1 Introduction

Open source software is a growing phenomenon in health information technology. While there has been more or less level funding for public health in the Federal budget over the past several years, public health agencies (PHA) continue to struggle with limited funding for information technology (HIT) projects (as well as overall). Software licenses can often account for a significant portion of HIT project cost and any savings there can often be applied to much needed funding for staffing.

To date the Centers for Disease Control and Prevention (CDC) which provides much of the funding for public health activities in the United States has shown limited serious interest in open source software as a way to reduce costs for their state, local, and tribal health partners. This attitude is far from ubiquitous at CDC, and there are several important projects that are based on open source projects, including the Reportable Conditions Knowledge Management System (RCKMS) being developed in partnership with the Council of State and Territorial Epidemiologists (CSTE).[i] Open source solutions – especially electronic health records (EHR) – have already been shown to be beneficial in low-resource settings.[ii]

Meanwhile, the Federal Government overall has taken several steps to make open source software a larger fixture within its technical environment. In 2016 the Chief Information Officer of the United States issues a new source code policy that established a goal of 20% of all Federally-developed code to be released into the open source community.[iii] Additionally, a repository was established to facilitate the migration of Federally-developed code to the public.[iv] While this addresses the potential migration of source code *from* the Federal government into the open source community it does not address *use* of open source software *by* the Federal government or its state, local, territorial, and tribal awardees.

There is already some evidence about the superiority of the fit of open source software in the healthcare market over proprietary products.[v] While there have been significant advances in open source software development and products available to public health, there appears to be much confusion around this topic related to open source licensing, management, fair use, and implied cost. This article will provide needed definitions of open source and proprietary software; describe the strengths, weaknesses, opportunities, and threats (SWOT) around each; provide examples of key open source products in use outside of public health; provide examples of a health open source market developing within one public health domain (immunization); and provide a case study of open source management and community which may serve as an exemplar of how an open source perspective can fuel a healthy alternative to the proprietary software market.

2 Laying the Ground Work: Some Definitions

First, some definitions are in order. Table 1 contains key definitions related to any discussion of software ownership and licensing:

Table 1 – Common Terms and Definitions

Term	Definition
Copyright	"…a legal right created by the law of a country that grants the creator of an original work exclusive rights to its use and distribution, usually for a limited time. The exclusive rights are not absolute; they are limited by limitations and exceptions to copyright law, including fair use."[vi]
Software copyright	"…the extension of copyright law to machine-readable software. While many of the legal principles and policy debates concerning software copyright have close parallels in other domains of copyright law, there are a number of

	distinctive issues that arise with software."[vii]
Public domain software	"…software that has been placed in the public domain, in other words there is absolutely no ownership such as copyright, trademark, or patent. Unlike other classes of licenses, there are no restrictions as to what can be done with the software. The software can be modified, distributed, or sold even without any attribution."[viii]
Copyleft (a play on the word copyright)	"…is the practice of offering people the right to freely distribute copies and modified versions of a work with the stipulation that the same rights be preserved in derivative works down the line."[ix]
Proprietary software	"…is licensed under legal right of the copyright holder, with the intent that the licensee is given the right to use the software only under certain conditions, and restricted from other uses, such as modification, sharing, studying, redistribution, or reverse engineering."[x]
Open source software	"…refers to a computer program in which the source code is available to the general public for use and/or modification from its original design."[xi]

The key concept here is the right to use and/or modify a software product. Copyright law in a country defines the rights of a creator of a software product. The owner/creator of proprietary software typically extends the right to use the software to another individual or organization based on payment of a one-time license fee or recurring right-to-use payment. Open source software is typically made available by its owner/creator to users *without* a license fee or other payment subject to specific terms and conditions that are identified in the open source license.

Public domain software was popular in the early days of computing but has become a somewhat misused term. Public domain software has *no* license and is therefore free to be used by anyone for any purpose, unlike open source software which *does* have a license, though that license may be fairly permissive in its terms and conditions. But not all "free" software is public domain (it may still have a license attached, just no fee for use). Copyleft licensing is typically used to make sure a modified piece of software is not then converted into a commercial product with restricted access or use. Pretty much all Copyleft products are open source, but clearly not all open source products are released under a Copyleft license.

Some additional clarification is necessary around basic concepts in computer software. Source code is the human-readable coding (or pseudo coding) that programmers write. Source code is then compiled (transformed) into machine code which computers can then execute. Only machine (executable) code is typically available to end-users, but open source licenses typically include the human-readable source code (thus the term open "source") as well as the executable code. Note that not all source code has to be compiled to be included in a license (open source or otherwise); some source code is "interpreted" by the computer at the moment the user executes it but is nonetheless exposed to the user to view and inspect. The most popular example is Hypertext Mark-up Language, or HTML, which is visible for inspection by any user through a web browser but which is interpreted in real-time by the web browser to render content on the screen. HTML code is still considered source code even though it can be visually inspected by any user.

Proprietary software can also be referred to as "closed source" because the source code is generally not made available to the user/licensee in order to protect the intellectual property of the owner/creator from undesired (or in some cases

uncompensated) modification or theft. Open source licenses typically include the right for the user/licensee to inspect and even modify the source code to understand or change the software's functionality. The Open Source Initiative (OSI) has identified as series of precise guidelines in its definition of open source which are detailed in Table 2.[xii]

Table 1 – Open Source Definition from the Open Source Initiative

1. Free Redistribution – no restriction on selling or giving software away, and no fee
2. Source Code – must be included, as well as compiled form, without fee
3. Derived Works – must be allowed, with distribution under same terms
4. Integrity of The Author's Source Code – can require that modifications are distinguishable from the original (*e.g.*, different version number)
5. No Discrimination Against Persons or Groups
6. No Discrimination Against Fields of Endeavor (*e.g.*, business use, or research use)
7. Distribution of License – included with the software
8. License Must Not Be Specific to a Product – rights transfer even if software parsed or repackaged
9. License Must Not Restrict Other Software that might be distributed with it
10. License Must Be Technology-Neutral (*i.e.*, no particular technology dependence)

As open source license varieties have proliferated over the years, OSI has initiated an approval process to identify licenses that are compliant with its definition to try to reduce the confusion surrounding license terms.[xiii] Popular licenses are identified in Table 3. Some licenses permit downstream commercial development (*e.g.*, Berkeley Software Distribution or BSD); some require contributions back to the originator (*e.g.*, GNU General Public License or GPL) - each has benefits and challenges and their selection is based on the objectives of the particular open source project.

Table 3 – Popular Open Source Licenses

Name	URL
Apache	https://www.apache.org/licenses/LICENSE-2.0
GNU General Public License	http://www.gnu.org/licenses/licenses.html
BSD	https://opensource.org/licenses/BSD-3-Clause
MIT	https://opensource.org/licenses/MIT
Mozilla Public License	https://www.mozilla.org/en-US/MPL/

Open source is an easing of the default copyright for software. The open source concept is about right to modify source code as well as the right to use software. Many license variations and conditions are possible. Open source can promote sharing, but also inhibit sharing through potential loss of intellectual property rights. Mixing open source and proprietary products can have important impacts on a software developer and a software project and must be done carefully and intentionally.

3 Comparison of Proprietary and Open Source Licensing Attributes and Impacts

For reasons cited above, PHAs have particular interest in containing cost for software. Open source solutions may offer particular benefits to PHAs. Mockus *et al* developed hypotheses about

open source product development versus proprietary product development and tested those hypotheses by studying two large open source projects, Apache and Mozilla.[xiv] Crowston *et al* conducted an exhaustive literature review of open source software development processes, identifying both challenges and opportunities.[xv] The following analysis examines the Strengths, Weaknesses, Opportunities, and Threats (SWOT) of both proprietary software and open source software strategies to help PHAs identify when it might be advantageous to use each approach assuming software solutions are available in each category for the particular functional requirement.

3.1 SWOT Analysis: Proprietary Software

Table 4 provides details on the strengths, weaknesses, opportunities, and threats to public health in using proprietary software. Note that this analysis applies to both the public health *application* market as well as the market for common productivity applications like work processing or spreadsheet tools.

Table 4 – SWOT Analysis, Proprietary Software

Strengths • Source code remains unified as only vendor/owner makes changes • Vendor, not user, bears the burden of enhancements • Software support usually easy to acquire from vendor, vendor-authorized partners, or even third parties • Software is usually relatively mature and well tested	**Weaknesses** • License fee charged by vendor to use software • Only vendor can make changes to the software • Potential loss of access to source code (and therefore continuing improvements and bug fixes) if developer stops working on the product or ceases operations entirely • Users may or may not get the enhancements they want based on vendor priorities and timetable • May or may not enable modular system deployment as interfaces to proprietary products may themselves be proprietary and not open
Opportunities • Depending on the program, CDC or another government agency may leverage its own funds and facilitate product-specific enhancements • Vendors of these products may more readily support external hosting as a way to provide additional revenue streams for themselves and reduce their own support and maintenance costs	**Threats** • Small public health software market may see even fewer vendors over time • Vendor reaction to encroachment of open source on its market is hard to predict • Agency funding continues to be constrained

Users of proprietary software often have a more reliable source of products, support, and enhancements, but that comes at a price, sometimes even a steep price. These users also run the risk that vendors of proprietary software will not make improvement, enhancements, or

corrections in a timely or even functionally-acceptable way. In the public health market specifically the cost of proprietary software may be prohibitive given limited and often shrinking budgets. The size of the market for specialized public health application software may also constrain the availability of these products and threaten the ability of PHAs to find affordable products that meet their missions.

3.2 SWOT Analysis: Open Source Software

Table 5 provides details on the strengths, weaknesses, opportunities, and threats to public health in using open source software.

Table 5 – SWOT Analysis, Open Source Software

Strengths	**Weaknesses**
• No license fee to use software • No loss of access to source code (and therefore continuing improvements and bug fixes) if developer stops working on the product or ceases operations entirely • Freedom to make/share changes with other agencies, organizations, or users • More transparency in product governance as open source development usually involves more clearly disclosed priorities and decision-making processes • Enables modular system deployment as interfaces are themselves open • Participative development allows many eyes to scan source code for possible errors or security exposures	• Risk of detrimental source code "forking" increases as anyone with access to the source code can make changes that may be inconsistent with the original product • Burden of enhancements may fall to individual users/organizations as there may be no central owner to perform or even coordinate this work • Software support may be harder to acquire as there may be no central owner to perform or even coordinate this work • Security patches may be slow to reach users as responsibility for updates may be more diffuse than proprietary software
Opportunities	**Threats**
• Collaborative development among PHAs or even other types of users can reduce cost of enhancements and support • Commercial vendors often provide solid support even for products they do not own • Move to more modular systems might enable more open source component use	• Cloud-based services offering open source product access for a fee effectively make these products behave as proprietary solutions often with no contribution back to the open source community • Public health community will usually not financially support product development • Public health community expects open source market to behave like commercial market by providing functional improvements to software without charge • Commercial vendor reaction to encroachment of open source on its market is hard to predict

Users of open source software agree to forgo the potential for more reliable vendor support and accountability by saving the cost of the one-time or ongoing license fee which can be prohibitive. The savings can often be substantial. Open source software users gain a modicum of control by being able to participate in the decision-making processes of open source developers and

by voting with their dollars (or their own programming resources) to implement in the software the features *they* want.

Additional factors can also muddy the waters. Often, open source products are extended with additional features and repackaged and sold as new products, sometimes referred to as "open core." While these new products are usually developed within the terms and conditions of the licenses of the underlying open source products, the open source version is often used by some vendors as a loss-leader to drive commercial sales with limited support. Even more serious, as noted as a threat in Table 5, cloud-based services offering open source product access for a fee effectively make these products behave as proprietary solutions often with no contribution back to the open source community. Some have described this phenomenon as a crisis within the maturing open source market as vendors continue to look for ways to monetize open source products legally at the expense of the collaborative and "free" nature of these products originally.[xvi]

There are many examples of open source software in use today. Table 6 identifies some popular open source products and their function.

Table 6 – Popular Open Source Products

Product Name	Use
Firefox	Web browser
Linux	Major operating system
JBoss	Java application server
MongoDB	Document database
Moodle Virtual Learning Environment	Academic course management
OpenOffice	Desktop productivity
PostgreSQL	Relational database management system
Thunderbird	E-mail client
Wordpress	Used for websites and blogging

Additionally, there are many examples of open source software used in healthcare today, and some software specifically used within public health as well.[xvii] Table 7 identifies some commonly-used healthcare and public health products.

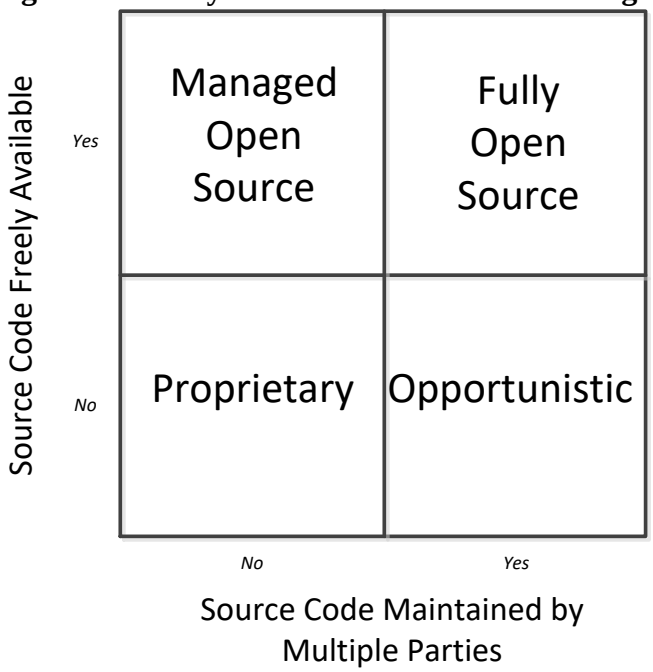
Table 7 – Selected Open Source Products Used in Healthcare

Product Name	Use
Choicemaker[xviii]	Patient matching
HAPI[xix]	HL7 v2.x message parser
Immunization Calculation Engine (ICE)[xx]	Immunization Evaluation and Forecasting
NextGen Connect (originally Mirth)[xxi]	Interface engine
OpenCDS[xxii]	Clinical decision support (CDS)
OpenMRS[xxiii]	Electronic Health Record System
R[xxiv]	Statistical computing

3.3 Policy Model for Software Licensing

Proprietary and open source software decisions are made in a complex landscape. It is helpful to think about licensing decisions along two dimensions as displayed in Figure 1: the horizontal axis identifies whether a product's source code is maintained by just one party or by multiple parties. The vertical axis identifies whether the product's source code is freely available.

Figure 1 – Policy Model for Software Licensing



Each of the four cells represents the possible intersection points of these two attributes. So, when source code is maintained (meaning, developed) by just one party *and* is *not* freely available we associate that with proprietary software as we have been discussing it (lower left quadrant of the matrix in Figure 1). When a product's source code is maintained by more than one party *and* is freely available we associate that with fully open source software as we have bene discussing it (upper right quadrant in Figure 1). But when a product's source code is maintained by just *one* party but the software *is* freely available we associate that with what we call *managed* open source (upper left quadrant in Figure 1). And, finally, when software is maintained by multiple parties but *not* freely available we call the *opportunistic* software as the outcome of the collaboration is restricted to the participants alone (lower right quadrant in Figure 1). We will revisit some of these distinctions in the next section.

4 Case Study: Open Source Community and Management of Expectations

PHAs have a tendency to believe that as government/non-profit agencies they should not be expected to pay for software and may feel no responsibility to assist in funding the development or support of open source products they may use. On the other hand, PHAs have a strong tradition of collaboration, often facilitated by numerous membership organizations to which they or their staff members belong. Open source software development and use is often managed formally in an open source *community*. The community is a self-selected group of individual users and organizations who collaborate on an open source software project. The benefits and

limitations of this community approach to software development are well documented.[xxv] Not all open source projects operate this way, and it is more common when the community of users is relatively small, but there are exceptions, including Linux which operates this way with a very large number of participants.[xxvi]

The focus is on collaboration in software development, testing, and support. Different application developers are encouraged to work independently on source code "forks," or variants, and there is a process for proposing that newly-developed code be made part of the core product while ensuring both its quality and compatibility with the existing software. The most successful open source communities offer a paradox: collaboration and experimentation with source code is encouraged, but the integration of new code into the production source code is fiercely controlled. In some projects, though, there may be barriers to participation that if managed properly can be overcome.[xxvii]

For a typical, more specialized open source product that might be developed within healthcare or even public health specifically, the management of user expectations may be difficult. In 2012, HLN Consulting (HLN) began developing an immunization evaluation and forecasting system initially for use in the New York City Citywide Immunization Registry (CIR) as its clinical decision support (CDS) module for immunization called the Immunization Calculation Engine, or ICE.[xxviii] The project began as contracted work for hire but with NYC's permission was moved into the open source world by its release under the GNU Lesser General Public License (LGPL) version 3.[xxix] ICE is a web service deployed using OpenCDS, a general purpose open source CDS product. It evaluates a patient's immunization history for clinical efficacy and then forecasts any immunizations that may be due now or in the future. A default set of clinical rules based on clinical guidelines developed and published by a Federal advisory committee, are distributed with the web service which is in use today in public health and clinical settings. HLN maintains and distributes the source code and run-time code for the web service as well as the default rules.

A number of interesting challenges emerged as ICE became more broadly used, including:

- As an open source product, ICE is neither commercial (*i.e*, responsive to the market) nor custom developed (*i.e*, responsive to its funders).
- Without rigorous control over software development and implementation there is potential for confusion and error through the publication and distribution of variations in the source code, though users are free to "fork" the product within the license terms.
- Even with general consensus over the clinical guidelines documented on the ICE wiki, some users might not agree nor accept consensus decisions. ICE does support multiple rule sets.
- Some users may not want to participate in a consensus process, especially if they are paying an organization (like HLN, the software's originator) for support.
- Some users may be interest in a subset of the rules (for instance, adult or childhood) to the exclusion of others and this may affect their interest in more broad participation in product decisions.
- Management and coordination of a collaborative process requires purposeful effort and funding to be sustainable.
- There is a key equity issue: who pays for enhancements when everyone benefits in the end?

In order to control any uncertainty that users and prospective users of ICE might experience, HLN developed and promulgated a set of *principles* to guide its ongoing development and support of this product:

- Changes to the open source software should be available to all users.

- HLN (or anyone else) may create products with "enhanced features" that must comply with the open source license but might not be freely available.
- A base set of rules developed by consensus should be maintained and be freely available to all users.
- Alternate rule sets may or may not be freely available at the discretion of the organizations that create them or sponsor their creation.
- Resources and activities should be leveraged across participants as much as possible.

As an open source software developer, HLN is not obligated to provide any more participation by its community of users in decision-making around ICE than it chooses to. More restricted participation may result in users (or prospective users) "voting with their feet" and opting out of using the product. Excessive participation may result in demands from the user community that have no realistic means of being accomplished or which interfere with HLN's own experienced perception of the direction that the product should go. To provide a balanced solution, corresponding to the Managed Open Source option in Figure 1 above, HLN implemented a two-tiered advisory structure to allow ample user feedback and participation while leaving the ultimate decision-making about ICE features and functions to HLN (and the marketplace, of course). First, HLN leveraged and expanded a Subject Matter Expert (SME) work group which existed since the product's inception. Work group members were initially drawn from the PHAs and organizations that were originally involved in ICE's creation. Its members are primarily clinical or public health experts who help tactically interpret clinical guidelines and their impact on ICE clinical rules. As more user organizations began to show interests in ICE and even deploy it within their own systems, the SME WG was carefully expanded to allow broader participation subject to HLN's agreement.

Additionally, HLN realized that more strategic advice about product direction and priorities. In the fall of 2017 HLN convened a Review Board made up of a small set of invited stakeholders from across the ICE user community: PHAs, electronic health record (EHR) vendors, academics. The Review Board meets quarterly to advise HLN but has no authority over decisions that HLN might make about product direction or features. HLN considers the advice of the Review Board through the lens of its principles described above.

Advice for Public Health Agencies (and Others)

So what advice can we offer to public health when it comes to open source software? PHAs should leverage widely-used, general-purpose open source products in public health systems where feasible (*e.g.*, Linux, PostgreSQL, HAPI, NextGen Connect). In addition, PHAs should explore jointly developing and supporting more specialized products with sister agencies or other organizations when necessary (*e.g.*, patient patching products, CDS, data quality assurance tools). From a practical standpoint, PHAs should embrace a Managed Open Source model whenever practical, and avoid the temptation to restrict participation with more Opportunistic approaches (Figure 1).

Agencies should look beyond the public health community for collaboration partners and public health and the general healthcare communities share many common goals and needs (*e.g.*, EHRs, personal health records or PHRs). They should encourage one organization to maintain stewardship over and support each product to prevent "detrimental" forking (Managed Open Source from Figure 1), and recognize and try to manage any turbulence these actions may cause in the commercial product marketplace. This may involve honest and open discussion with commercial vendors about their place in an agency's environment and the changing role of open source.

References

[i] Surveillance/Informatics: Reportable Condition Knowledge Management System. https://www.cste.org/group/RCKMS. Accessed October 28, 2019.

[ii] Syzdykova a, Malta a, Zolfo M, Diro E, Oliveira J, Open-Source Electronic Health Record Systems for Low-Resource
Settings: Systematic Review. *Journal of Medical Internet Research.* 2017;5(4):e44.

[iii] M-16-21: Memorandum for the Heads of Departments and Agencies. https://sourcecode.cio.gov/. Published August 8, 2016. Accessed October 28, 2019.

[iv] Sharing America's Code: Unlock the tremendous potential of the Federal Government's software. https://code.gov/. Accessed October 28, 2019.

[v] Reynolds C and Wyatt J. Open Source, Open Standards, and Health Care Information Systems. *Journal of Medical Internet Research.* 2011;13(1):e24.

[vi] Copyright. https://en.wikipedia.org/wiki/Copyright. Accessed October 28, 2019.

[vii] Software copyright. https://en.wikipedia.org/wiki/Software_copyright Accessed October 28, 2019.

[viii] Public-domain software. https://en.wikipedia.org/wiki/Public_domain_software. Accessed October 28, 2019.

[ix] Copyleft. https://en.wikipedia.org/wiki/Copyleft. Accessed October 28, 2019.

[x] Proprietary software. https://en.wikipedia.org/wiki/Proprietary_software. Accessed October 28, 2019.

[xi] Open source. https://en.wikipedia.org/wiki/Open_source. Accessed October 28, 2019.

[xii] The Open Source Definition (Annotated): Version 1.9. http://opensource.org/osd-annotated. Accessed October 28, 2019.

[xiii] The License Review Process. https://opensource.org/approval. Accessed October 28, 2019.

[xiv] Mockus A, Fielding R, Herbsleb J. Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology*. 2002;11(3), 309–346.

[xv] [xv] Crowston K, Wei K, Howison J, Wiggins A. Free/Libre open-source software development: What we know and what we do not know. *ACM Computing Surveys*. 2012; 44(2), Article No. 7.

Open source confronts its midlife crisis. http://dtrace.org/blogs/bmc/2018/12/14/open-source-confronts-its-midlife-crisis/. Published December 14, 2018. Accessed October 28, 2019.

[xvii] Karopa T, Schmuhl H, Demski H. Free/Libre Open Source Software in Health Care: A Review. *Healthcare Informatics Research.*2014: 20(1), 11-22.

[xviii] Open Source Record Matching. https://www.choicemaker.com/. Accessed October 28, 2019.

[xix] Hapi: Welcome. https://hapifhir.github.io/hapi-hl7v2/. Published June 23, 2017. Accessed October 28, 2019.

[xx] Open-Source Immunization Forecasting Software. https://www.hln.com/ice. Accessed October 28, 2019.

[xxi] NextGen® Connect Integration Engine. https://www.nextgen.com/products-and-services/integration-engine. Accessed October 28, 2019.

[xxii] Open Clinical Decision Support (OpenCDS) Tools and Resources! http://www.opencds.org/. Accessed October 28, 2019.

[xxiii] The global OpenMRS community works together to build the world's leading open source enterprise electronic medical record system platform. https://openmrs.org/. Accessed October 28, 2019.

[xxiv] The R Project for Statistical Computing. https://www.r-project.org/. Accessed October 28, 2019.

[xxv] Kogut B and Metiu A, Open Source Software Development and Distributed Innovation. *Oxford Review of Economic Policy*. 2001: 17(2), 248-264.

[xxvi] Linux.org. https://linux.org/. Accessed October 28, 2019.

[xxvii] Sholler D, Steinmacher I, Ford D, Averick M, Hoye M, Wilson G (2019) Ten simple rules for helping newcomers become contributors to open projects. 2019: *PLoS Comput Biol* 15(9): e1007296.

[xxviii] For a more thorough discussion of the ICE project see Suralik MJ and Arzt NH, Anatomy of a Public Health Open Source Project: HLN's Immunization Calculation Engine (ICE). *Open Health News*, January 29, 2019. http://www.openhealthnews.com/articles/2019/anatomy-public-health-open-source-project-hlns-immunization-calculation-engine-ice. Accessed October 28, 2019.

[xxix] Licenses. http://www.gnu.org/licenses/licenses.html. Published December 15, 2018. Accessed October 28, 2019.

