Open Source Primer: What Public Health Needs to Know

American Medical Informatics Association (AMIA)
June 13, 2017

Noam H. Arzt, PhD, FHIMSS
*President, HLN Consulting, LLC*
The American Medical Informatics Association is accredited by the Accreditation Council for Continuing Medical Education to provide continuing medical education for physicians.

The American Medical Informatics Association designates this live activity for a maximum of 1 *AMA PRA Category 1 Credit(s)*. Physicians should claim only the credit commensurate with the extent of their participation in the activity.
Disclosures

The following planners disclose they have no relevant financial relationships with commercial interests:

Working Group webinar planner: Christina Stephan

AMIA staff: Susanne Arnold

Content Reviewer: Seth Foldy

Presenter Noam Arzt discloses that he is a stockholder in HLN Consulting, LLC.
Learning Objectives

After participating in this activity, the learner should be better able to:

- Identify key differences between Open Source and proprietary software
- Identify how open source software might better (and more cheaply) support/enhance the use of information technology to improve services and data management
- Implement strategies that can be employed within public health to leverage Open Source offerings
- Understand the notion of the Open Source "Community" and its usefulness to public health.
Agenda

- Introduction and definitions
- Open Source “Community”
- SWOT for Open Source and Proprietary
- Examples from Outside of Public Health
- Public Health Examples: IIS World
- Strategies for Success
- Case Study: Open Source Governance
- Resources
Why is this an issue for Public Health?

- Continued funding uncertainty
- Increased CDC interest
- Increased Federal interest overall
- Ongoing development of tools and products
- Confusion over the topic – need guidance on licensing, governance, fair use, and cost implications
"Copyright is 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."

https://en.wikipedia.org/wiki/Copyright

"Software copyright is 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."

https://en.wikipedia.org/wiki/Software_copyright
“Public domain software is 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.”

https://en.wikipedia.org/wiki/Public_domain_software

Popular in the early days of computing but often a misused term. “Free” software may or may not actually be public domain. Public domain products have no license.
Definitions

“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.”

https://en.wikipedia.org/wiki/Copyleft

Typically, this is 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 not the other way around.
Definitions

“Generally, **open source** 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.”


**Source code**: What programmers write

**Machine code**: What computers understand

Source code is typically (but not always) compiled (transformed) into machine code which users can then execute. Only machine (executable) code is typically available to end-users, but open source includes the human-readable *source* code.
Definitions

**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)

http://opensource.org/osd-annotated
Definitions

“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.”

https://en.wikipedia.org/wiki/Proprietary_software

Generally source code is not made available. Can also be referred to as “closed source.”
Definitions: Summary

- Over the years, open source license varieties began to proliferate
- OSI initiated an approval process to identify compliant licenses to try to reduce confusion
- Popular licenses:
  - Apache
  - GNU General Public License
  - BSD
  - MIT
  - Mozilla Public License
- Some licenses permit downstream commercial development (*e.g.*, BSD); some require contributions back to the originator (*e.g.*, GPL) - each has benefits and challenges.
Definitions: Conclusions

- Open source is an easing of default copyright for software
- Open source concept is about right to modify source code as well as the right to use software
- Many variations and conditions 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
Open Source Community

- Not all projects operate this way
- More common where the community of users is smaller, but there are exceptions (e.g., Linux)
- Focus on collaboration
- Benefit from the knowledge of others: many project encourage others to “fork” the software and make their own modifications
- Most successful model offers a paradox: collaborative development but fierce control of the “production” source code
**SWOT Analysis: Proprietary**

<table>
<thead>
<tr>
<th><strong>Strengths</strong></th>
<th><strong>Weaknesses</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Source code remains unified</td>
<td>• License fee to use</td>
</tr>
<tr>
<td>• Vendor bears the burden of enhancements</td>
<td>• Only vendor can make changes</td>
</tr>
<tr>
<td>• Software support usually easy to secure</td>
<td>• Potential loss of access to source code if developer stops work</td>
</tr>
<tr>
<td>• Software is relatively mature and well tested</td>
<td>• Users may or may not get the enhancements they want</td>
</tr>
<tr>
<td></td>
<td>• May or may not enable modular system deployment</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Opportunities</strong></th>
<th><strong>Threats</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• CDC IDIQs/BPAs may fund and facilitate product-specific enhancements</td>
<td>• Small PH software market may see fewer vendors over time</td>
</tr>
<tr>
<td>• Vendors of these products may more readily support external hosting</td>
<td>• Vendor reaction to encroachment of Open Source is hard to predict</td>
</tr>
<tr>
<td></td>
<td>• Agency funding continues to be constrained</td>
</tr>
</tbody>
</table>
# SWOT Analysis: Open Source

<table>
<thead>
<tr>
<th><strong>Strengths</strong></th>
<th><strong>Weaknesses</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• No license fee to use</td>
<td>• Risk of <em>detrimental</em> source code “forking”</td>
</tr>
<tr>
<td>• No loss of access to source code if developer stops work</td>
<td>• Burden of enhancements may fall to individual users/organizations</td>
</tr>
<tr>
<td>• Freedom to make/share changes</td>
<td>• Software support may be harder to secure</td>
</tr>
<tr>
<td>• Transparency in governance</td>
<td></td>
</tr>
<tr>
<td>• Enables modular system deployment</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Opportunities</strong></th>
<th><strong>Threats</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• “Joint development” can reduce cost of enhancements &amp; support</td>
<td>• PH community will not financially support product development</td>
</tr>
<tr>
<td>• Commercial vendors often provide solid support</td>
<td>• PH community expects open source market to behave like commercial market</td>
</tr>
<tr>
<td>• More modular systems might enable more Open Source component use</td>
<td>• Commercial vendor reactions</td>
</tr>
</tbody>
</table>


Examples from Outside of PH

- Linux: Major operating system
- WordPress: Used for blogging
- FireFox, Thunderbird: Web & E-mail
- OpenOffice: Desktop productivity
- PostgreSQL: Relational Database Management System
- Moodle Virtual Learning Environment (VLE): Course management
Example: Immunization Information System

- CAT Quality Assurance Tool
- Choicemaker (patient matching)
- Data Quality Assurance Tool (DQA)
- FEBRL (patient matching)
- HAPI (HL7 Parser)
- Immunization Calculation Engine (ICE)
- Mirth (Interface Engine)
- Texas Children’s Hospital Forecaster
Strategies for Success in PH

- Begin to move Public Health systems to modularity and Service-oriented Architecture (SOA)
- Leverage widely-used Open Source products *where feasible* (*e.g.*, Linux, PostgreSQL, HAPI, Mirth)
- Jointly develop/support more specialized products when necessary (*e.g.*, forecaster, QA tools)
- Look beyond public health community for collaboration (*e.g.*, EHRs, PHRs)
- Encourage one organization to maintain stewardship over and support each product to prevent “detrimental” forking (Managed Open Source)
- Recognize and manage any turbulence this may cause in the commercial product marketplace
Case Study: Open Source Governance
Background

- ICE is an immunization evaluator/forecaster originally developed for use in an Immunization Information System.
- Accepts patient DOB, gender, immunization history, disease immunity and produces an evaluation and forecast.
- Standards-based (web services, HL7 vMR)
- NYC agreed to release products into Open Source community under GNU Lesser General Public License (LGPL) version 3
Governance Challenges

- Because ICE is open source, it is primarily but not exclusively responsive to those who contribute staff or financial resources.
- Although users are free to “fork” the product, without strong management of the software development effort there is great potential for confusion & error.
- Even with consensus over the CDS rules some users might not agree nor accept consensus decisions.
- Management and coordination of governance process requires purposeful effort and funding to be sustainable.
- Equity issue: who pays for enhancements since everyone benefits in the end?
Governance Principles

- Changes to the Open Source software should be available to all users.
- 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.
- Anyone may create products with “enhanced features” that must comply with the Open Source license but might not be freely available.
New ICE Governance Process

- Goals
  - Implement more participative way to govern rule updates
  - Support ongoing ICE software development & maintenance

- Establish a Review Board to
  - Provide an unbiased review and validation of ACIP recommendations as they are proposed for implementation in ICE
  - Review relevant proposed enhancements to product functionality

- Subject Matter Expert Workgroup to
  - Support the Review Board
  - Make the specific detailed decisions regarding the rules for the ICE default immunization schedule.
New ICE Governance Process

- Strive to maintain a “core” or “base” rule set.
- Continue to post openly on the wiki
- Plan and post a product roadmap
- Maintain unified control of the actual contents of the software distributed and maintained by this collaborative effort.
Possible Funding Models for Enhancing Open Source Software

- First one who needs a new feature pays for it
- Negotiate joint/collaborative funding to implement common needs
- Use “unexpended” support dollars (if sufficient)
- Seek outside funding from a supportive non-user of the software
- Combination of all of the above
Resources

- Open Source Initiative
- Open Source Electronic Health Record Alliance (OSEHRA)
  [https://www.osehra.org](https://www.osehra.org)
- OSEHRA License Terms Document
  [https://www.osehra.org/sites/default/files/osehra_licensing_terms_v.1.0.pdf](https://www.osehra.org/sites/default/files/osehra_licensing_terms_v.1.0.pdf)
- Draft Federal Source Code Policy
  [https://sourcecode.cio.gov/](https://sourcecode.cio.gov/)
Evaluation

- Please complete the evaluation you receive within the hour via email
Noam H. Arzt
President, HLN Consulting, LLC
858-538-2220 (Voice)
858-538-2209 (FAX)
arzt@hln.com
http://www.hln.com/noam/
@noamarzt