Intellectual Property for Global Goods: An OpenLMIS Case Study

By Josh Zamor, Kim Couri and Brandon Bowersox-Johnson - July 30, 2021

OpenLMIS is a leading Open Source electronic logistics management information system purpose-built to manage health supply chains in low- and middle-income countries. Serving over 11,000 health facilities in 9 countries in Africa, OpenLMIS manages commodities across all major health programs including vaccines and COVID-19 products.

Introduction

Open Source practitioners are used to endless debate about which Open Source license to choose, but we often do not consider the full suite of intellectual property (IP) elements. Open Source Global Goods have an important role to play in global health and development, therefore it is critical for anyone developing, reinvigorating, stewarding or interoperating Global Goods to understand the basics of Open Source IP. With this understanding, investments in software can be protected, contributions can be appropriately attributed and the experience of those interacting with Global Goods can be elevated. These things combined ultimately enable stronger product impact and sustainability.

This case study identifies five distinct, yet interrelated, elements of IP. These elements include: inbound contributions, upstream dependencies, trademark, copyright and license. OpenLMIS initiated this investigation as part of a long-term vision and planning process for a private sector partnership with Vitalliance that would support OpenLMIS' sustainable growth. We knew any partner's proprietary software would need to work smoothly with our OpenLMIS Open Source software. We set out to define IP parameters so that OpenLMIS could grow as an Open Source Global Good with a thriving implementer Community, and also enable a new partner to offer additional features and services.

Identifying clear practices and expectations around IP is important for a community to effectively manage its code contributions and build trust among contributors, as well as protect its software investment. Reviewing IP elements may be important for Global Goods as they embark upon new partnerships or for any key transition such as: starting a new project, major contributors joining or leaving or defining a community vision or plans.

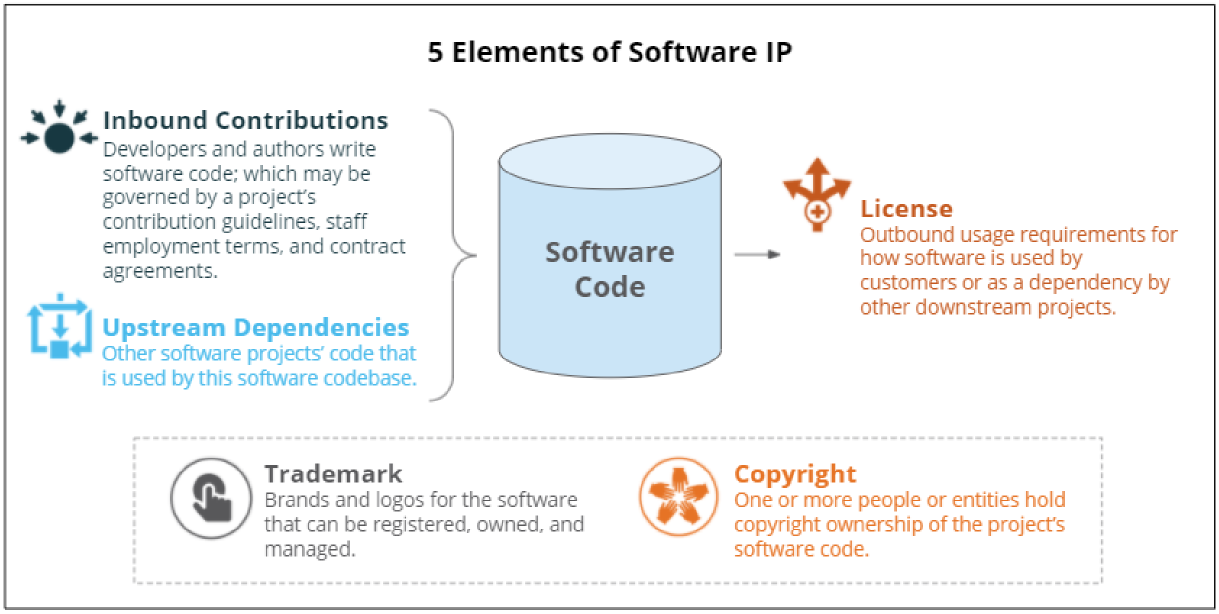
In this document, we walk through the options that OpenLMIS considered for each of the five IP elements, and share the path that we ultimately selected for each. We hope that sharing our journey can inform other Global Goods as they consider approaching a similar IP exercise.

Objectives and Approach

Our primary objectives in evaluating IP practices were to ensure alignment with the OpenLMIS Community Principles while also enabling a private sector partnership to create additional interoperable proprietary software. We carefully considered OpenLMIS' past experience and future goals to develop the following criteria to inform our decision making:

- **Public Health Use:** OpenLMIS must continue to be an Open Source Global Good available for use by current and future public health implementations.
- **Customization & Community Shared Value:** The OpenLMIS Community must still be a shared value Global Good, allowing code contributions to improve and enhance the system, while also allowing modular customization and extension.
- **Sustainable Business Models:** OpenLMIS must be compatible with a variety of Open Source business models and connectors to commercial platforms to unlock more functionality and options for sustainable revenue.

The OpenLMIS team evaluated five elements of software IP, shown in Figure 1, to determine how collectively each element influences software ownership, usability and shareability. We conducted our evaluation through desk research, Community input and conversations with Open Source experts and legal advisors. We document the options considered for each of the five elements and explain which option we ultimately chose for OpenLMIS.



Inbound Contributions Developers and authors write software code; which may be governed by a project's contribution guidelines, staff employment terms, and contract agreements.

Upstream Dependencies Other software projects' code that is used by this software codebase.

License Outbound usage requirements for how software is used by customers or as a dependency by other downstream projects.

Trademark Brands and logos for the software that can be registered, owned, and managed.

Copyright One or more people or entities hold copyright ownership of the project's software code.

Figure 1: 5 Elements of Software IP

Inbound Contribution Policy

Inbound contribution policy guides the copyright ownership of any code or work developed for an Open Source project. When a developer/author writes code or creates work, including documentation, by default, the copyright is held by the author. However, if the author has an employment agreement the copyright is typically assigned to their employer. For example, as of 2021, VillageReach, who has stewarded OpenLMIS since the creation of version 3 in 2016, has a contract with a software partner, SolDevelo, to provide OpenLMIS software development. This means for any works developed by SolDevelo, VillageReach owns the copyrights.

Inbound contributions can range from informal practices to formal documents. The more formal options can have added benefits that allow the copyright holder to relicense the work. Formal documentation provides the most flexibility, but it also discourages casual code contributions (i.e. those without a formal employment relationship).

This is particularly relevant if an Open Source project engages with an institution such as a university or college. Many Global Goods have cooperation with academic institutions, who often have strict rules about retaining copyright and IP rights for any work done by students, faculty and staff.

Open Source projects tend to handle inbound contributions from developers/authors in the following ways:

1. **Copyright Assignment:** Each author submits code and assigns the copyright for their contributions to the project's copyright holder. This can happen directly from author to project, or indirectly through multiple levels of employment agreements and contracts.

2. **Contribution License:** Each author submits code but retains the copyright for their contributions and grants the project a permissive license to their work. This allows the project to incorporate their work and make it available under the outbound license (see License section below). Usually the inbound license and outbound license match. This is a default approach used by many Open Source projects, including projects on GitHub. An Open Source project does not need a legal agreement for this option.

3. **Contributor License Agreements (CLA):** Each author and/or their employer has a signed CLA that grants the project a license to their work. A CLA is a signed agreement that each contributor, either an individual or organization, must sign before contributing code. CLAs can contain language that limits project flexibility related to relicensing the work. (In other words, the inbound license can limit the choices of the outbound license.)

4. **Contributor Assignment Agreements (CAA):** Each author and/or their employer has a signed CAA that assigns the copyright of their work to the project. CAAs are a routine and formal approach similar to the Copyright Assignment option (#1 above). Like a CLA, a CAA is a signed agreement that each contributor (individual or organization) must sign before they submit code. In a CAA, you give the Open Source project ownership (copyright) instead of just giving the project a license to use an author/organization's contribution.

5. **Developer Certificate of Origin (DCO):** Each author submits code along with a light-weight DCO, and the author retains the copyright for their contributions and grants the project a permissive license to their work. The DCO approach is most similar to the CLA (#3 above), but it is an easier process for developers to submit a

DCO rather than having to review and sign a longer legal contract that an Open Source project must keep on file.

6. **Ad-Hoc/Undocumented:** Open Source projects may not have a specific, documented approach. Over time, as people come and go, as pull requests, forks and commits happen, the exact documentation of all IP contributed may remain undocumented. That said, even if it is not documented, copyright automatically exists whenever work is created.

OpenLMIS Result: **In early 2021, OpenLMIS decided to evolve OpenLMIS' inbound contribution policy to a formal primary CAA and alternatively a CLA for approved contributions. The CAA and CLA make contributions more explicit. The CAA ensures that a small group of parties holds the IP with clear documentation for all contributions. This makes it cleaner for the OpenLMIS Community to evolve ownership and licensing in the future. The alternative CLA may be granted upon special approval should the contributor require that they retain their copyright in exchange for project flexibility in relicensing the contribution.**

Upstream Dependencies

Most Open Source projects use lots of other Open Source libraries and components to build their project. OpenLMIS v3 uses Java Spring, AngularJS and hundreds of Java and Javascript libraries that are part of the build and compilation process. These upstream dependencies are documented in the OpenLMIS wiki as of August 2020.

The OpenLMIS v3 policy is to only use upstream dependencies that have one of the approved Open Source Initiative (OSI) licenses. If the outbound license used for OpenLMIS were changing from AGPLv3, it would be important to confirm that our upstream dependencies were compatible with our new license.

OpenLMIS Result: **OpenLMIS is continuing to use upstream dependencies only if their license is one of the approved OSI licenses.**

How Copyright and Open Source Licenses Work Together

Copyright and licenses are two IP elements that are uniquely intertwined. Internationally copyright is held, automatically, by the person/organization that creates the work. It is common that a person's employment contract takes ownership of their work, or the owner assigns the work to another person or organization. As soon as there is an established owner (via a copyright), the work may not be used by others (non-owners) unless a specific license is granted. To do this, the Open Source movement has created many different licenses, which generally allows others to use copyrighted work.

Copyright

The inbound contribution policy defines how the Open Source project gains copyright of any work contributions (e.g. source code, documentation, etc.). This section addresses how copyright is subsequently held by the Open Source project.

Most software projects hold copyright in one of two ways:

1. **Unitary Copyright:** One person or legal entity holds the copyright for the project.
2. **Multi-Party Copyright:** Two parties or more can hold copyright for the project. For instance, in the Linux kernel, thousands of individuals and companies hold the copyright for their contributions. Within a multi-party copyright arrangement, there is a spectrum of permissions around how rights may be granted back to the other copyright holder(s). In an extreme case, multi-party arrangements may be considered a "collective work" and/or may make it impractical to get all parties to agree to any license changes.

OpenLMIS Result: **In the OpenLMIS Community, VillageReach and John Snow, Inc. hold the OpenLMIS v3 copyright. Other organizations and contributors may join with a formal CLA. The OpenLMIS Community will continue to explore longer-term options to transfer copyright ownership to a neutral fiscal sponsor in the future. Choosing a copyright holder or fiscal sponsor has lots of considerations beyond the scope of this document, including 501(c)(3) benefits and compatibility with donor funding mechanisms.**

License

There are many different Open Source licenses. It is important to research which option is right for an Open Source Project, based on its advantages and disadvantages. OpenLMIS has a strong copyleft license called AGPLv3, which was initially selected to encourage Community contributions and user freedom through the "network effect." (This case study will not delve into the debates around relationships between Open Source licenses and the costs of free software.)

It is worth noting that the comparison of options below have been specifically chosen and evaluated for OpenLMIS, and its new private sector partnership. A broader comparison of license options can be found in the GitHub Choose a License Repository.

1. **Remain AGPLv3:** In this scenario, OpenLMIS would remain AGPLv3 and not issue any other licenses.

    - *Advantages:* There would be a number of advantages to keeping the current license, which most notably is the likelihood of a more seamless transition. Furthermore, the private sector partner would be able to use the extension architecture to build proprietary services (i.e. with their own license) that work with the existing OpenLMIS' services.

    - *Disadvantages:* In terms of disadvantages, the AGPLv3 license lessens the incentive for customers of the private partner to buy licenses for the current codebase or a direct derivative work. The AGPLv3 also has a "network effect" that some SaaS vendors dislike, as it requires the disclosure of the source, including any derivative, to the end user.

2. **Relicense:** OpenLMIS could be relicensed under a more permissive license (e.g. MIT or OSL).

    - *Advantages:* A more permissive license may be more business "friendly". For example with an OSL license, the partner could make a derivative work of an existing OpenLMIS service and sell a license for a closed version of it without disclosing their modifications.

    - *Disadvantages:* It may be confusing to the new partner where their derivative work begins and OpenLMIS' ends. It could also be used by anyone to distribute a closed version of OpenLMIS without disclosing the source. At the far end, this could create an incentive for implementations to not share enhancements, and instead compete against the new partner, impacting community sustainability goals.

3. **Custom License:** Similarly to the scenario above, OpenLMIS could be relicensed under a more permissive license and then the current copyright holder grants a separate license to the new partner.
    - *Advantages:* This opens up options for different types of secondary (or tertiary) licenses to be issued.
    - *Disadvantages:* Having multiple licenses may be confusing to the Community and could discourage contributions if only a single partner would be granted a right to profit from the code.
4. **Dual License:** This is a popular approach that can allow a single Open Source project to be available in both a Community Edition and a Commercial Edition.
    - *Advantages:* With a dual license, the copyright holder could retain an Open Source license for the project, while allowing for closed-source derivatives and proprietary licenses to be sold.
    - *Disadvantages:* This would allow the project to be relicensed to a more permissive Open Source or other license without the obligation to share changes. This would allow any entity to fork the project and sell licenses for custom development. This could also be potentially confusing to the Community.

OpenLMIS Result: **OpenLMIS remains AGPLv3 and is not issuing any other licenses at this time.**

Trademark

Upon review, we found that the OpenLMIS logo and mark were governed by a common law trademark and we identified that a formal registration could better protect the OpenLMIS brand. The logo and OpenLMIS brand were being used consistently by the Community of Trusted Partners to promote OpenLMIS services. OpenLMIS projects sometimes also use country-specific names and apply their own logo onto the software, which fits our Community norms.

With Open Source products, trademarks can be an important mechanism for the community to control behavior such as forking. For example, trademark rights can be used to elevate official products or to police "inferior" products or those that take the product in a different direction.

OpenLMIS Result: **OpenLMIS can file a federally-registered trademark and this documentation will help in the future if OpenLMIS IP is ever moved to a neutral fiscal sponsor (see Copyright section above). Trademark registration is effective based on its date of first use rather than its filing date, but it is still important to track the origins of any brand names and logos and to retain documentation of first use.**

Conclusion

In formalizing these five IP elements we ensured OpenLMIS can continue to grow and serve its user base in public health systems, while also taking advantage of new opportunities through our new private sector steward. We recommend that any Open Source Global Good consider all five elements, not just licensing alone, when planning for its future. It's especially important to review these at key moments such as starting a new Open Source project, publishing a new piece of code, changing major contributors, or evaluating the project's long-term goals. By thoughtfully managing IP and making decisions with Community stakeholders, OpenLMIS is confident that we are well-positioned to be sustainable and continue to thrive for years to come.

A special thanks to Mark Wittow, Destinee Evers, and Jonathan Shallow of K&L Gates LLP; McCoy Smith of Lex Pan Law; and VM Brasseur for your guidance and thought leadership that contributed to our journey.

Please note that the views expressed in this document are intended for informational purposes only. The content has been specifically curated for OpenLMIS and to share amongst other interested parties; it is not a substitute for legal advice.