Releasing Tools for International Disease Surveillance as Open-Source Software: A Case Study

Raj J. Ashar

Since the development of tools for the Suite for Automated Global Electronic bioSurveillance (SAGES) began in 2008, the SAGES team and sponsor have envisioned the eventual release of these tools as open-source software to the global public health and technology communities. Open-source software allows members of the public to study, customize, and operate their own local copies of the software and source code, often without monetary fees. As such, releasing SAGES as open-source software assures prospective users that they retain complete control over the health data collected by SAGES-based systems, and aligns well with the model of self-sustainability intended for the operation of SAGES systems in resource-limited settings. Preparing two SAGES tools, OpenESSENCE and SAGES Mobile, for release as open-source software projects entailed a multifaceted, months-long effort that spanned policy, technical, and community considerations. This article describes the issues, trade-offs, and decisions that were addressed leading up to the successful open-source release of OpenESSENCE and SAGES Mobile in June 2013. The aim of this case study is to inform future Johns Hopkins University Applied Physics Laboratory (APL) and external efforts to release open-source software.

INTRODUCTION

Open-source software follows a design methodology that makes software source code available to members of the public for usage, modification, reproduction, and study. This methodology stands in contrast to that for commercial software, whose source code may be considered a trade secret and is often unavailable to the public. The transparency afforded by open-source software can empower end users in several ways. It can build the user's trust that the software is operating as advertised, without surreptitiously performing any unintended operations

on the user's data. Additionally, open-source software can strengthen an end user's sense of system ownership by allowing the user to customize and operate the code in a manner that satisfies the user's requirements. It should be emphasized that releasing open-source software entails much more than providing the source code for a software application to an individual sponsor or partner organization.

The source code for an open-source software tool is generally made available in accordance with a license agreement, which is a framework of terms specifying rights and obligations for every party to the software. In many cases, open-source software developers make their tools publicly available without charging users fees for procuring or operating the software. This means that a plethora of open-source software tools for numerous purposes are freely available to anyone who has sufficient hardware and network resources to run the software. Consequently, free open-source software can dramatically lower a system's total cost of ownership and make such systems particularly suitable for resource-limited settings.[1]

Since the development of the first tools for the Suite for Automated Global Electronic bioSurveillance (SAGES) in 2008, the SAGES team at the Johns Hopkins University Applied Physics Laboratory (APL) has envisioned the eventual release of these tools as open-source software to the global public health and technology communities, with the long-term goal of transitioning SAGES development and evolution to those diverse communities.[2] This vision has been shared by our sponsor, the Global Emerging Infections Surveillance and Response System, a division of the U.S. Armed Forces Health Surveillance Center (AFHSC-GEIS, http://www.afhsc.mil/geis). AFHSC-GEIS recognized that developing freely available open-source SAGES tools could invest its international partner organizations more deeply in the success of deploying public health surveillance systems based on SAGES software. The open nature of the software would assure international partners that they retain complete control over the SAGES systems and the data being collected, and the free cost of software would allow partners to dedicate scarce resources toward other public health priorities. It was hoped that these advantages together would allow SAGES-based systems to be deployed and self-maintained at minimal cost to all parties involved, and to become self-initiating and self-sustaining efforts that provide international partners with greater insight into the health of their populations for years to come.

International deployments of the SAGES tools began in 2009 and have taken place across Africa, Asia, and South America.[2,3] These deployments demonstrated that SAGES enhances public health situational awareness in resource-limited countries.[3,4] The introduction of SAGES in these settings has succeeded in large part because countries can independently collect, analyze, and control the dissemination of population health data within their own national borders.[4] As the initial goals of developing sustainable and customizable electronic disease surveillance systems were realized, interest in SAGES and requests for technical assistance started to grow.[4]

In addition to professional interest in SAGES among the global public health and technology communities, some teams working at prospective overseas sites expressed interest in obtaining the SAGES source code so they could verify that the software stores population health data entirely within their own national borders. Additionally, individual SAGES developers who had supported deployments at different sites began fielding e-mail requests for assistance, which complicated the team's efforts to provide a timely, coordinated, and centralized response to common inquiries. To fulfill the sponsor's stated direction of furthering "the development of SAGES as an open-source tool that can be installed and configured without direct U.S. support," the impetus for releasing the SAGES tools as open-source software became clear by the end of 2012.[5] Hence, the SAGES open-source release was scheduled to take place at the end of June 2013.

Successful open-source projects, such as the Mozilla Firefox web browser, are sustained over time by community interest in, and dependence on, the software.[6] As such, the responsibilities of providing major enhancements, maintenance patches, publicity, technical support, and documentation eventually come to rest with a group of contributors who personally use the software—a group that is much larger than the original development team.[7] Although this community interest parallels the forces of market demand that sustain other types of commercial software, the open nature of these projects enables a comparatively more nimble pace of improvement and innovation.[8] Thus, the open-source model aligns well with the long-term vision for SAGES.

Although any new open-source project could potentially reuse a significant amount of existing open-source licenses, tools, and infrastructure, it will still contend with an assortment of choices and challenges.[9] To distribute software to a wide audience, a project needs to select and integrate components that are compatible from both a technical standpoint as well as a licensing perspective.[6] Once a project "goes live," motivating volunteers to remain involved becomes a managerial task.[7] Open-source projects that originate from private industry or government organizations may need to further satisfy institutional policy and quality requirements before release. APL project teams that intend to release their work as open-source software face distinct challenges because of APL's roles as a university-affiliated research center and a trusted agent for the United States government (USG), which are codified in numerous legal, contractual, and procedural requirements.

Notwithstanding the challenges posed by the task of making software open source, the advantages of contributing software to the community have been recognized by the USG. Because there are circumstances under which releasing open-source software serves the government's interests, the USG has already made numerous contributions to the open-source community.[6,10] One software project funded through the DoD, the Ozone Widget Framework (OWF), was even directed by congressional mandate "to publish and maintain on the public Internet the [OWF] application programming interface specifications, a developer's toolkit, source code," and other resources, and to "establish a process by which private individuals and companies may voluntarily contribute" improvements to source code, documentation, and the underlying application programming interface.[11]

At APL, there is precedent for making significant contributions to open-source software that benefit sponsors, end-user communities, and the public at large. Under the former Global Engagement Department, independent research and development funds allowed APL to implement and release key components of the .NET version of the NASA World Wind geospatial visualization tool (http://worldwind.arc.nasa.gov/index.html). Sponsor funding has enabled the Asymmetric Operations Sector (AOS) to develop and release major new features for the Xen hypervisor (http://www.xenproject.org/developers/teams/hypervisor.html) and the OpenStack cloud computing platform (http://www.openstack.org/).[12] The release of the FEAT Editor (http://sourceforge.net/p/feateditor/wiki/FEAT%20Editor%20Home/), a tool developed with sponsor funding by the National Security Analysis Department to edit federation agreements for simulations, as a stand-alone open-source project was expedited through contractual language that specified explicitly that open-source software should be developed while also maintaining compliance with USG acquisition and export control policies.[13]

This article describes the multifaceted effort to release two SAGES tools, OpenESSENCE and SAGES Mobile, as open-source software that can be independently obtained, customized, and operated. The release of these tools marked the first APL open-source software contribution to the global public health and technology communities. In addition to the basic decisions that must be made for every new open-source software project, the nature of this open-source release posed two fundamental challenges. One challenge was to satisfy various institution-specific requirements for ongoing public releases of technical work, while the other challenge was to make the SAGES tools readily accessible and usable for a technically diverse international audience. It is hoped that this article serves as a guide for future APL and external project teams that seek to make the fruits of their labor publicly available as open-source software.

METHODS

Approach

Early in 2013, the SAGES software development team began the effort to release SAGES as open-source software. At that time, versions of both OpenESSENCE and SAGES Mobile had already enjoyed several years of operational deployment.[2,4,14] SAGES software team members collectively share decades of software engineering, open-source, and international deployment experience. Based on the team's expertise, three types of end users were identified as the target audience for the SAGES open-source release:

1. **Information technology (IT) liaisons:** Overseas IT staff who maintain and troubleshoot deployed SAGES systems
2. **Public health end users:** Overseas public health staff and epidemiologists who rely on SAGES systems for regular public health surveillance
3. **Independent users and developers:** SAGES users and software developers who are not directly affiliated with APL or the sponsor but who obtain SAGES via public Internet download and potentially contribute code to SAGES

One SAGES software engineer was tasked with leading the open-source release effort, in close collaboration with other software team members and SAGES project management. APL resources outside the project were consulted as needed for approvals and project support. These APL resources included AOS leadership, the Office of Counsel, the Business and Communication Services Department, the Information Technology Services Department, and the APL OpenStack software development team. In particular, consulting with the OpenStack team helped provide direction for the SAGES open-source release because their project already makes contributions to the open-source community and thus had worked through the various policy requirements.

Open-Source Release Considerations, Risks, and Opportunity

Releasing open-source software encompasses three broad categories of considerations: policy, technical, and community. Policy considerations pertain to the statutory, contractual, software licensing, and institution-specific requirements that an open-source software project must satisfy for the purpose of minimizing risk to all stakeholders. Technical considerations pertain to all aspects of engineering and distributing the software openly, from the stages of design, implementation, and testing, to release, packaging, and operational deployment.[7] Community considerations pertain to building and engaging an end-user community in order to ensure

that the software remains as accessible and useful as possible. Although some aspects of these considerations may not be apparent to end users, holistically addressing all three categories is vital to the success and longevity of an open-source project. A further explanation of the tasks and decision points involved with each consideration is provided in the results below.

Risk is inherent to these considerations and to the overall endeavor of releasing open-source software. Although there are numerous websites where source code can be posted under the terms of an open-source license, there is no "cookie-cutter" template of processes and tools that will guarantee a project's success. In fact, one study of more than 170,000 projects hosted on SourceForge found that the majority of open-source software projects fail before reaching at least three software releases.[15] There are numerous ways a project can place itself at greater risk of failure by not effectively addressing aspects of one or more considerations. If policy issues are not resolved, this may force the open-source project to abruptly change or cease its activities, which affects existing end users who rely on the project for patch upgrades. Policy issues could further subject the project's sponsors, development team, and independent developers to reputational harm, civil liability, or even criminal liability. Intractable technical issues adversely impact the quality and functionality of released software, which will in turn drive away prospective end users and developers. In the absence of a community that depends on the project, technical support for the software will be scarce, and there will be few enthusiastic end users to market the software by word of mouth.

That said, effectively addressing all three categories of considerations does not by itself ensure a project's success. Rather, long-term end-user adoption and adaptation of the software is key, but that is accompanied by its own risks. If the community takes interest in the software, it is likely that the software's functionality, underlying source code, and supporting artifacts will then be subjected to a greater degree of scrutiny and critique than previously experienced, which may make the original development team feel as though it is under siege from criticism.[7] Regardless of the causes, the failure of an existing open-source project will strand end users, who have often invested significant time and resources integrating the software into their workflows, because it leaves those users without future security updates, technical support, and new features to improve the existing software. The consequences of a project failure may also indirectly affect stakeholders who rely on the software but do not use it directly, such as decision makers who depend on the software for critical information.

Despite the risks inherent to releasing open-source software, the strengths that SAGES already has, including the distinct niche it fills, and the potential benefits stemming from the widespread adoption of SAGES present a compelling opportunity for an open-source success that enhances global public health. SAGES leverages IT advances to improve public health surveillance capabilities in resource-limited countries, which can aid in the early detection of, and response to, disease outbreaks of international concern.[2,4] The SAGES sponsor has recognized the critical importance of SAGES software for maintaining international health security and has thus continued to support this work over several years. Members of the SAGES team welcome the release of SAGES to the open-source community and are focused on constantly improving its quality and usefulness to end users. SAGES has built an existing user base across parts of the world over the last few years.[2–4,14] Interest in SAGES continues to mount, with teams in some prospective host countries expressing the desire to verify that SAGES does indeed store their health data within their national borders. Therefore, the release of SAGES as open-source software will allow the global public health and technology communities to deploy public health surveillance capabilities in new settings, either independently or with sponsor support. Because lessons and improvements will be identified through these new deployments, the release also brings the project closer to accepting external technical contributions from public health liaisons and independent developers.

RESULTS

Overview

An overview diagram of the steps involved with the SAGES open-source software release process is provided in Fig. 1. Although technical activities are at the heart of the SAGES open-source release process, there are interdependencies between the policy and technical considerations and between the technical and community considerations. Fortunately, some activities could be executed in parallel so that the release process proceeded more efficiently.

Policy Considerations

Policy considerations need to be addressed carefully in order to minimize legal and reputational risks to the software developers, their organization and sponsors (if applicable), and software end users. In brief, the concerns underlying this category of consideration boil down to two root questions:

1. Can the software be released as open source?
2. If the software can be released as open source, what are the terms under which release would be permissible?

An open-source project in the United States must comply with numerous obligations that are imposed by

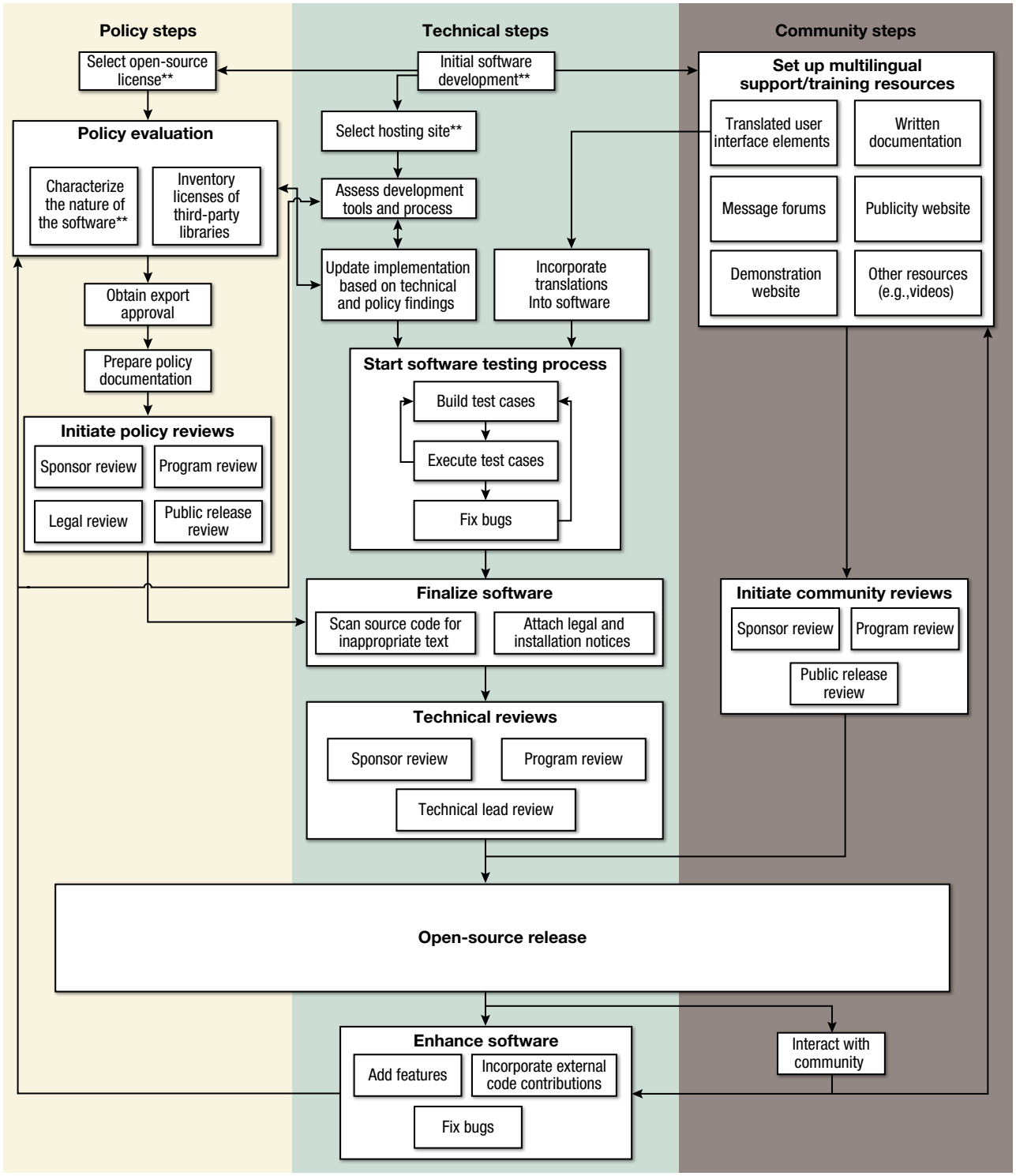

Figure 1. Illustration of the SAGES open-source software release process at APL.[17] The three columns of this process map outline the steps involved with each of the policy, technical, and community considerations. The vertical axis of this map shows the sequence of steps that must take place during the release process. Steps marked with asterisks (**) normally take place only when a software tool is first made open source.

federal statute, funding sources and home institution (if applicable), and the third-party software on which it depends. (For the purposes of this discussion, written obligations imposed by third-party software fall under the category of software licensing obligations. Additionally, obligations that are not formally codified in a contract document fall under the category of institutional obligations.)

During this release effort, it became clear that statutory and institutional concerns were intertwined. Similarly, contractual and software licensing concerns were also found to be intertwined. The two sets of related concerns are discussed below.

Statutory and Institutional Obligations

From a statutory perspective, export regulations governing the release of software to non-U.S. entities vary depending on whether the software has military applications. If the software does not have military applications, its export is still prohibited to any embargoed country, to any end user who is deemed to be of concern, or in support of any prohibited usage activity.[13] Noncompliance with export regulations may subject stakeholders of an open-source project to criminal or administrative penalties.[16] Based on the statutory considerations, the nature of SAGES needed to be characterized early in the process in order to determine whether the open-source release should proceed.

For the SAGES open-source release, additional procedural concerns came into play. These concerns centered on embracing the open-source paradigm while minimizing risk to APL and to the sponsor. At APL, risk mitigation often involves consulting with, and obtaining approvals from, internal public release, export control, and legal experts. APL project management and the sponsor then provide oversight on an ongoing basis. In turn, these requirements dictate a need to establish formal release processes and guidelines. Throughout the release process, the SAGES team worked closely with APL experts and the sponsor to address public release and export control concerns.

Several policy decisions fundamentally shaped the SAGES open-source release. First, addressing the question of whether the release could even go ahead was crucial. Sponsor agreement for releasing the SAGES tools as open-source software had been in place from the project's inception and was reconfirmed before the release.[17] Early in the open-source release process, SAGES was granted internal export authorization to proceed under the condition that its functional purpose remains as a "health data collection, analysis, and reporting platform."[17] This approval established the basis for the release effort to proceed.

Agreement also had to be reached regarding the actual public release process that the SAGES team would follow. SAGES project management received sponsor concurrence for a proposal that met USG needs for public release review while also accommodating the dynamism inherent to the open-source community. That proposal made a distinction between major and minor software releases of functionality. Before a major release, a release notes document describing the capabilities of the release will be provided to the sponsor for review, and APL will execute structured code reviews and integration testing of those capabilities. A minor release, which may fix bugs or augment existing capabilities, will not require prior sponsor review because it falls under the scope of the documentation provided for the associated major release. Internally, SAGES was granted an exemption to improve the tools on an ongoing basis without continuously triggering the APL public release process.[17] With this exemption, the SAGES software development team shifted to using GitHub (the chosen hosting site, as described below) as the sole hosting site for APL and external development of core SAGES functionality.

These policy decisions were formally codified for future reference and may serve as a template for future APL efforts to release open-source software. Before the open-source release, an internal policy memorandum describing the release process, which incorporated documentation of key decisions, was distributed to APL stakeholders.[17] After the release, guidance for making appropriate public commits to GitHub, including a code review checklist, was issued to the SAGES software development team.[18]

Policy considerations directly influenced technical discussions and activities pertaining to third-party dependencies. Export concerns drove the close scrutiny of two libraries that feature encryption capabilities. One library, Spongy Castle (http://rtyley.github.io/spongycastle/), was proactively removed from the SAGES Mobile software stack because of its support for sophisticated elliptic curve cryptography.[19] Another library, Spring Security (https://github.com/spring-projects/spring-security), was kept in the OpenESSENCE software stack after confirmation that projects already in the public domain fall outside the scope of export control.[17]

Software Licensing and Contractual Obligations

Leaders of an open-source project must select the license terms under which the project asserts copyright. In turn, these terms must fully respect all contractual obligations imposed by the project's source(s) of funding. This consideration is particularly relevant in the context of open-source software whose development has been funded by the USG, such as SAGES. Government acquisition contracts typically invoke statutes that require that the USG retain "unlimited rights" to use, reproduce, and modify software developed at taxpayer expense.[6,13] Noncompliance with contrac-

tual obligations may subject an open-source project to civil penalties.

The selected license places obligations on the way end users can modify and distribute the software. One significant decision that an open-source project team must make is whether the project should be licensed under the terms of "copyleft" provisions. Essentially, this means that any work that is derived from a copyleft-licensed work must also be licensed under the same copyleft-license terms if that derivative work is itself distributed.[20] As such, copyleft licenses are also pejoratively referred to as "viral" licenses because the original terms of redistribution are automatically passed on to any software that incorporates copyleft-licensed code.[7] Versions 2 and 3 of the GNU General Public License (GPL) are among the most widely recognized examples of copyleft licenses, and at a minimum they require anyone who distributes modified versions of GPL-licensed software to also make available the modified source code to the recipients.[7, 20–22]

A project team's decision regarding whether to adopt a copyleft license extends upstream to any software components that the project incorporates. Open-source software projects like SAGES generally reuse components, or libraries, from third-party open-source software. Third-party libraries provide capabilities that have already been developed, and hopefully continue to be maintained, by the community. Open-source software projects built on these libraries are obligated to fully comply, or be compatible, with the terms of the libraries' licenses. If a library is licensed under terms that are not compatible with the license selected for the software project, then that library will need to be removed from the software and replaced before the software can be distributed. Given a copyleft-licensed library, a restrictive interpretation of the license terms may argue that a software application that merely depends on the library also becomes subject to the same copyleft terms.[7, 23] Consequently, an open-source project may elect to avoid distributing libraries licensed under copyleft terms, such as the GPL.[17]

Thus, the license terms of an open-source project must be compatible with third-party library licenses, as well as the statutory and contractual obligations outlined above. Numerous available resources describe the considerations that must be weighed when an open-source project evaluates different types of licenses for potential adoption.[7, 13, 24, 25] An analysis performed more than a year before the start of this open-source release effort had recommended the Apache License, Version 2.0 (hereafter referred to as "Apache 2.0") for OpenESSENCE.[17] As a license that is commonly adopted by open-source software projects, Apache 2.0 was selected in order to comply with third-party library licenses that OpenESSENCE linked to at that time, and to avoid imposing copyleft requirements on independent developers.[17, 26]

Unfortunately, finalizing the selection of the open-source license for all SAGES tools became complicated by the prior release of a SAGES tool under different terms. The selection of Apache 2.0 as the license for OpenESSENCE prompted questions regarding why the license agreement developed for an older version of a different SAGES tool, known as ESSENCE Desktop Edition (EDE), had not been adopted instead. (EDE is a single-user analysis and visualization tool for disease surveillance in resource-limited and disaster settings that has also been developed under the SAGES umbrella of tools. It is meant to be installed on a single computer and does not require an Internet connection. Because EDE is a mature tool that has been deployed successfully in the Republic of the Philippines, further development of EDE is not planned at this time.[2, 4]) That license agreement, which had been negotiated directly between APL and a specific deployment site, did not provide end users with all of the rights to modify and distribute code that are integral to open-source software. Ultimately, the SAGES team decided to distribute both OpenESSENCE and SAGES Mobile under the terms of Apache 2.0.[17]

Applying the Apache 2.0 license to SAGES subsequently required striking a balance between honoring contractual requirements and fully preserving end-user rights. An appendix to the Apache 2.0 license provides a standard license notice that should be attached to works licensed under Apache 2.0;[26] typically, this notice is copied to the top of all source code files as a header. Given that SAGES had been supported by USG funding, APL was contractually required to insert language stating USG rights in the software. In general, the boilerplate language generally used by APL to state USG rights in header notices is as follows: "This material may only be used, modified, or reproduced by or for the U.S. Government pursuant to the license rights granted under FAR clause 52.227-14 or DFARS clauses 252.227-7013/7014."[17] However, the SAGES team felt that including the APL boilerplate notice "as is" with the Apache header could be perceived as precluding non-USG usage of SAGES, which would contradict the intent of the open-source release. The APL Office of Counsel resolved this dilemma by updating the boilerplate USG rights language to read "This material may be used, modified, or reproduced by or for the U.S. Government pursuant to the rights granted under the clauses at DFARS 252.227-7013/7014 or FAR 52.227-14."[17]

A prerelease inventory was taken to confirm that SAGES relies on third-party dependencies whose licenses are compatible with Apache 2.0. That review first cataloged the libraries and corresponding licenses to which open-source distributions of OpenESSENCE and SAGES Mobile link, excluding any libraries that are used internally for testing purposes.[27, 28] (Because a binary-only version of EDE would be made available on the SAGES publicity website, a list of the librar-

ies and licenses to which EDE links was also compiled in the interest of due diligence.[29]) The list of licenses in use (see supplemental Table 1 at http://www.jhuapl.edu/techdigest/TD/td3204/32_04-Ashar-Tables.pdf) was compared to a list of licenses that the Apache Software Foundation has deemed "similar in terms" to Apache 2.0.[30] OpenESSENCE was found to rely on a handful of graphing libraries that were licensed under the terms of the GNU Lesser GPL (LGPL), which is not compatible with Apache 2.0. Nevertheless, those libraries could still be redistributed because OpenESSENCE only statically links to their binary versions and does not modify any underlying source code. Additionally, three libraries used by OpenESSENCE were found to be dual-licensed under both the terms of GPL and exemptions to GPL for software licensed under the terms of Apache 2.0; in those cases, SAGES qualified for the exemptions. In summary, the third-party libraries referenced and redistributed by SAGES are licensed in a manner compatible with Apache 2.0.

Technical Considerations

The technical considerations that arise when releasing open-source software center on preparing the software and project to gracefully handle greater usage, scrutiny, and external contributions from IT liaisons as well as independent users and developers. As such, there are two types of activities that form the technical basis for an open-source release:

1. Selecting the open-source hosting site where the software will reside
2. Undertaking activities to improve the quality and maintainability of the software, so that end users can download a complete "package" of functioning software for test and modification purposes

These activities are described in the following sections.

Hosting Site Selection

A key technical decision for the project is selecting the hosting site for source code and supporting materials. First and foremost, the target hosting site will serve as the project's technical repository, where developers across the world will go to download the source code and potentially contribute improvements back into the project. The heart of this repository is the version control system (VCS), which maintains an audit trail of all submitted changes to project files. Through this audit trail, anyone can look back at all prior commits and versions of the code base.

There are numerous VCSs available, with Subversion, Git, and Mercurial being among the most prevalent.[7] For an open-source project, one challenge is to select a VCS that will remain popular with software developers through the foreseeable future.[7,31] The hosting site may also feature bug-tracking capabilities, which can prove particularly useful when the project begins to accept bug reports and code contributions from independent users and developers. Given the importance of the hosting site to an open-source project, the project team may seek a site that shares similar objectives to their own project and that makes the project highly accessible and visible to the software's target audience.

To begin the search for a hosting site, the team began by understanding the existing technical composition of SAGES and identifying features desired for the upcoming open-source release. OpenESSENCE and SAGES Mobile are implemented primarily in Java and JavaScript on top of open-source software technology stacks. Source code for both tools had been available internally to APL developers through a Subversion VCS, and there was discussion on the SAGES software team about transitioning SAGES to the newer Git VCS. The desire for robust workflow and collaboration capabilities, as well as the software team's knowledge about the target user base, informed a survey to gather data for the selection of the hosting site.

The survey resulted in a thorough review of widely used open-source hosting sites in existence as of January 2013. This survey consisted of a number of criteria identified by the SAGES development team to ensure that the hosting site's features aligned with the workflow, visibility, and sustainability goals, as well as overall purpose, of the open-source SAGES project. Alphabetized lists of the survey criteria, including their definitions, are provided in the Appendix, available at http://www.jhuapl.edu/techdigest/TD/td3204/32_04-Ashar-Appendix.pdf. The following types of criteria span the categories of interest for the open-source release:

- **Technical:** Pertinent technical details about the hosting site
- **Workflow:** Support for collaboration, communication, and receiving code contributions from IT liaisons and independent developers. (Note that the communication aspect of this type of criteria overlaps with the community considerations described below. At the outset of this effort, it had been hoped that the hosting site could also host the message forums and project web pages.)
- **Visibility:** Availability and prominence of the software project to current and potential end users
- **Focus:** Whether the site's technical objectives aligned with those of SAGES. (For example, some hosting sites focused on technology stacks not used by SAGES, such as Microsoft .NET, or did not even focus on open-source software.)
- **Cost:** Cost for publicly hosting project software
- **Informational:** General information about the hosting site that did not enter into the selection process

The SAGES development team evaluated the candidate hosting sites against the survey criteria primarily on the basis of features advertised by their public web pages, including "Terms of Service" agreements and web pages for individual open-source projects hosted by the sites. In some cases, responses for the survey were obtained by creating a free account on a site and logging in to view its features. When responses to survey criteria could not be obtained through these means, the sites' sales or technical support teams were contacted for clarification; responses were received to most requests for information. (Note that the actual quality of features on each hosting site was not formally assessed by this survey.) The criteria themselves were divided into two sets: preliminary and main. The preliminary criteria were used to quickly eliminate candidate sites whose features clearly did not align with the requirements for hosting SAGES. If a site did not meet all of the preliminary criteria, then the site was eliminated from further consideration. The main criteria served to collect data about potentially viable hosting sites for the purpose of informing the selection process.

A list of 30 candidate open-source software hosting sites was compiled from several sources.[32–35] (Two hosting sites, GitHub and Savannah, were each evaluated twice because of the terms and conditions available to different types of site users.) Of the 30 candidates, 23 sites did not meet the preliminary criteria (see supplemental Table 2 at http://www.jhuapl.edu/techdigest/TD/td3204/32_04-Ashar-Tables.pdf). Main criteria data were collected for the remaining seven sites, including two sets of data captured for GitHub (see supplemental Table 3 at http://www.jhuapl.edu/techdigest/TD/td3204/32_04-Ashar-Tables.pdf). Based on the main criteria data, the SAGES software team narrowed the selection to GitHub (public repository) and Google Code. (For the purpose of discussion, GitHub [public repository] will hereafter be referred to simply as GitHub.)

Although these two sites lack desired features for communicating with a wider audience, such as public/private message forums, they satisfy core technical requirements and are popular with software developers. Both sites offer free hosting to open-source projects without displaying any advertising to end users. After considering the features of both sites, the SAGES software team elected to host SAGES on GitHub because of the enormous number of projects hosted by that site, as well as the site's perceived visibility in the developer community. At the same time, the team also decided to transition from a Subversion VCS to a Git VCS because of Git's growing popularity among developers as well as its robust support for distributed software development.

These decisions affected the technical direction of SAGES on several levels. Git allows developers to submit pull requests, which are essentially requests to review code changes before they are included in the trunk, or main code base. It also features the ability to easily make branches, or copies of an existing code base where developers can test adding their own enhancements for potential integration back into the trunk.[36] Thus, the selection of the Git VCS (http://git-scm.com/) anticipates eventually incorporating enhancements contributed by independent developers. GitHub envelops a community experience around Git because it allows developers to follow colleagues, monitor projects, discuss code, and submit issue reports.[37] Because open-sourcing software allows other developers to create entirely new tools from existing code, GitHub also makes forking a repository, or creating an entirely separate repository from an existing code base, very easy and traceable to the original code base.[38] Immediately after these decisions were made, developers on the SAGES software team quickly learned how to use Git; they have now been using it on GitHub without difficulties for more than a year.

Thus far, GitHub has provided a good home for OpenESSENCE and SAGES Mobile (Fig. 2). As of mid-February 2014, the SAGES GitHub page links to eight repositories of code for the various components that make up SAGES.[39] The SAGES software team relies on GitHub as the sole VCS for core SAGES functionality, as mentioned above. To date, the core OpenESSENCE repository is among the most active; it has seen more than 100 commits from eight contributors.[40] At the start of 2014, GitHub introduced a feature for monitoring traffic volumes to the different repositories, which we will use to gauge technical interest in SAGES.[41]

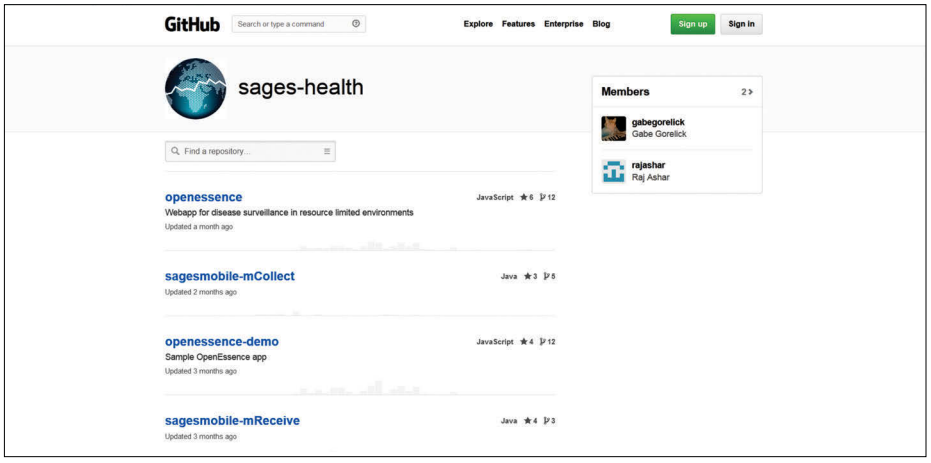

Figure 2. GitHub home page for SAGES software projects.[39]

Quality and Maintainability

The prospect of releasing the software openly to the world offers an opportunity to enhance its quality and maintainability. In addition to the quality assurance activities that normally occur in the course of software development, a period of rigorous testing and resolving critical bugs should take place before the release.[7] Approximately 2 weeks before the SAGES release, test cases for both OpenESSENCE and SAGES Mobile were developed collaboratively by SAGES public health experts and software developers on Google Drive documents. Approximately a dozen test cases were formally documented for SAGES Mobile, and several dozen test cases were developed for OpenESSENCE. These SAGES team members then executed the OpenESSENCE test cases using different versions of the Microsoft Internet Explorer, Mozilla Firefox, and Google Chrome web browsers. SAGES Mobile test cases were executed on a variety of smartphones running different versions of the Android operating system. During the testing period, 46 bug reports were submitted and addressed. Bugs found during testing were reported in a local instance of the JIRA software issue tracker (https://www.atlassian.com/software/jira).

The impending release also spurred the reevaluation of both code and tools that support the development process. Support for building multiple modules was enhanced in OpenESSENCE by migrating the project's build automation system from Maven (http://maven.apache.org/) to Gradle (http://www.gradle.org/). Additionally, manual scans of the code were performed to ensure that inappropriate text, such as the names of specific countries or developers, would not be included in the release. Although a small handful of instances were found and fixed, the preponderance of the code did not require any changes because writing code for eventual public release had been an objective throughout SAGES development.

Community Considerations

Cultivating a community of end users around open-source software involves first understanding the needs of the user base. For SAGES, the resources made available to the community have to accommodate a diverse audience. The technical skill sets of SAGES users primarily span the fields of public health and IT, while the cultural diversity of the user base spans multiple languages. In addition to building support for the multilingual user base directly into the software, the SAGES team stood up a slate of resources that allow end users to both independently obtain technical support and seek support from the developers and other users. Further details on both types of resources follow below.

Resources for Independent Use

Training resources that end users can reference at any time to help them use the software are essential to the success of open-source projects that seek longevity.[7] As a rule, written documentation is an essential part of making any publicly released software useful.[7] Existing English-language manuals for administering and using OpenESSENCE have been updated since the open-source release to reflect enhancements that have been added to the SAGES tools (http://www.jhuapl.edu/sages/support.html). Similar manuals for SAGES Mobile were provided after the inaugural release. Given that public health end users generally lack system administration skills, an OpenESSENCE demonstration capability had already been available over the web so that prospective end users could easily try interacting with a SAGES tool without first installing it. That public OpenESSENCE demonstration capability (https://openessence.jhuapl.edu/openessence; Fig. 3) has been periodically refreshed since the release, allowing anyone to practice using its analysis and visualization features.

Additionally, the intrinsically international nature of the SAGES user base required that the software itself feature user-interface support for non-English languages from the inaugural open-source release. To support the international user base, French and Spanish translations were built into OpenESSENCE (Fig. 4) and added to the software packages that constitute SAGES Mobile (Fig. 5). These "out-of-the-box" localizations will allow users in Africa, Latin America, and other parts of the world to immediately begin deploying SAGES.

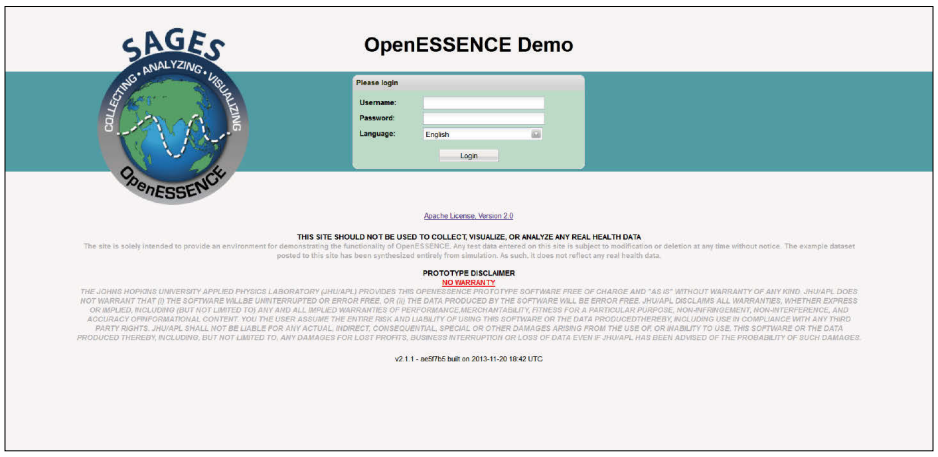

Figure 3. Log-in page of OpenESSENCE demonstration site.

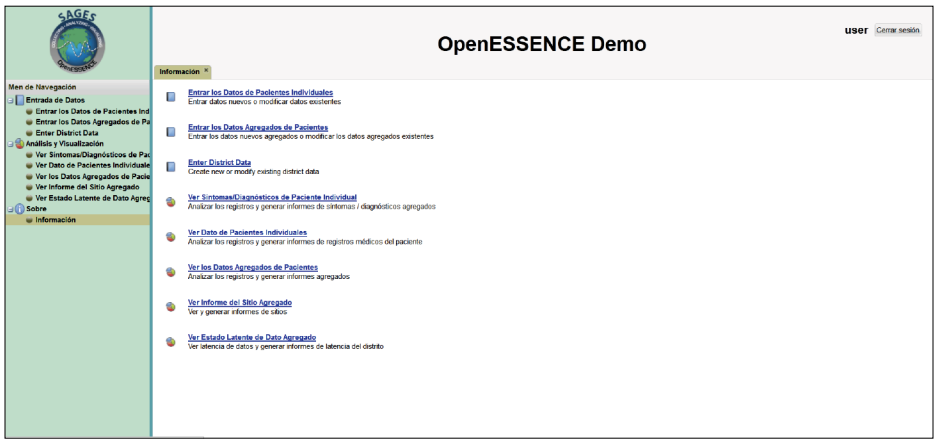

Figure 4. Spanish-localized home page of OpenESSENCE demonstration site.

Communication Resources

Multiple channels of communication need to be set up so that end users can learn about, and obtain technical support for, the software. These channels must be able to scale as interest in the software grows. Merely providing an e-mail address that allows users to contact development team members about issues can become burdensome quickly to developers and also impede end users from sharing best practices with each other.[7] For SAGES, the primary channels of communication that needed to be put into place were message forums and a publicity website.

Public message forums provide a means for end users to seek technical support, suggest new features and enhancements, and relate their experiences about using the software. Because the audience of the message forums includes the development team and end users, this medium offers a means for end users with common interests across the world to connect with each other. Moreover, this medium makes support requests visible to the entire development team, which allows questions to be answered in a coordinated fashion. Contemporary message forums typically feature archiving of messages, so a searchable knowledge base of questions and discussions about the software builds up over time. Because each in-country SAGES deployment has some degree of customization to meet local requirements, the ability to also create private message forums for addressing deployment-specific concerns was desired.

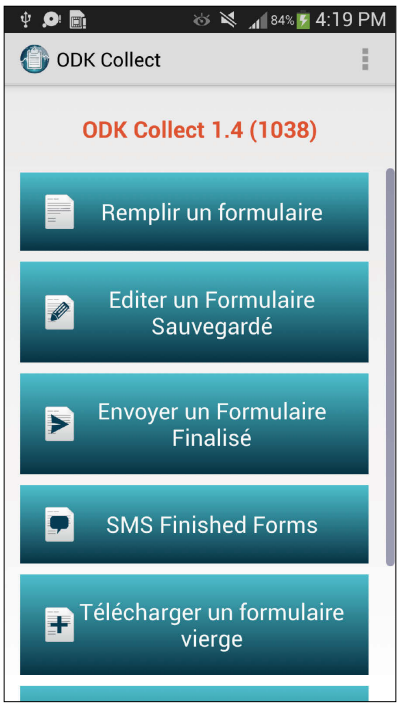

Figure 5. French-localized menu in SAGES Mobile.

Although it had originally been hoped that the hosting site would offer message forum capabilities, the survey found that the message forums offered by GitHub and Google Code did not feature all of the capabilities for mailing lists, private forums, and public forums desired for communicating with the different segments of the SAGES user base. An informal research effort in January 2013 looked at using the commercial vBulletin software (http://web.archive.org/web/20130118224503/http://www.vbulletin.com/) for message forums but found that it had the disadvantage of requiring that the SAGES project self-host and maintain it. Instead, Google Groups was adopted as the message forum solution for SAGES.

Google Groups freely hosts public and private message forums without showing advertising to forum subscribers. Google Groups offers a mailing-list feature, so subscribers are able to monitor and interact with forum discussions via e-mail, in addition to logging in with a web browser. One minor disadvantage of choosing Google Groups is that message-forum subscribers are required to possess Google accounts. However, that requirement was judged to be a minor trade-off in exchange for its overall flexibility and feature set. Three groups were initially set up on Google Groups. Two of those groups are intended for discussion among end users and developers, with the first group targeted to public health end users and the second group intended for technical discussion about SAGES.[42,43] A third group has been created primarily as a mailing list for announcements about SAGES, and it is targeted to a broad audience. As of February 2014, 53 subscribers around the world who had learned about

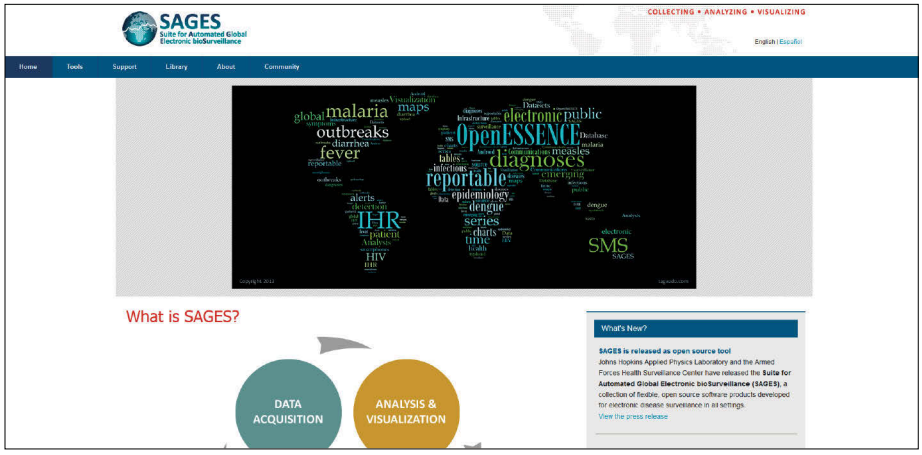

Figure 6. Home page of new publicity website for SAGES.

SAGES through an APL press release[3] or word of mouth had registered for the third group.[44]

To market SAGES across the world, a publicity website had been developed well before this open-source release effort began. The audience for this website includes both SAGES end users and others who are interested in SAGES but may not use it directly. (This other segment of the audience includes policy makers, the media, and members of the general public.) The team felt strongly that maintaining a publicity website was essential to the continued adoption and success of SAGES. Because the open-source release offered an opportunity to rethink the branding of SAGES itself, this meant refreshing both the content and the look and feel of the website (http://www.jhuapl.edu/sages/; Fig. 6) in concert with the open-source release.

The refreshed publicity website presents a curated view of information about SAGES and electronic disease surveillance, including lists of the third-party open-source libraries used by the SAGES tools. It also serves as a project hub: it links end users to the SAGES GitHub repositories and Google Groups and allows users to sign up for e-mail announcements about SAGES. The website itself makes OpenESSENCE, SAGES Mobile, and EDE executables available for download and deployment.[4] Through the website's integration with three popular social networks (Fig. 7), end users can spread a word-of-mouth message about SAGES to friends, colleagues, and the world. Shortly after the inaugural open-source release, a Spanish-language version of the website was established. The site was also adapted for optimal viewing on both mobile devices and PCs by applying a responsive web design approach.[45] To draw public attention toward SAGES and the milestone of this inaugural open-source release, a joint press release was issued by APL and the sponsor.[3]

The publicity website has been instrumental to the success of the SAGES open-source release. It won both approval and acclaim from the sponsor before the release. Since the release, web traffic to the publicity site has provided one means to gauge interest in SAGES across the globe. (The site's web traffic is being tracked by IBM Unica NetInsight v8.6.0.0 web analytics software, http://www-03.ibm.com/software/products/en/on-premise-web-analytics/.) Between the June 2013 public release of the site and mid-February 2014, there were more than 5,000 visits to the site with approximately 24,000 page views. Within those figures, hits to the Spanish-translated portions represented 79 visits and 222 page views. Visits to the website (based on Internet Protocol addresses) have originated from countries throughout all of the continents except Antarctica (Fig. 8). Approximately one-half of the publicity site's visits originated outside the United States, with traffic from China and India collectively representing more than 10% of the site's visits.

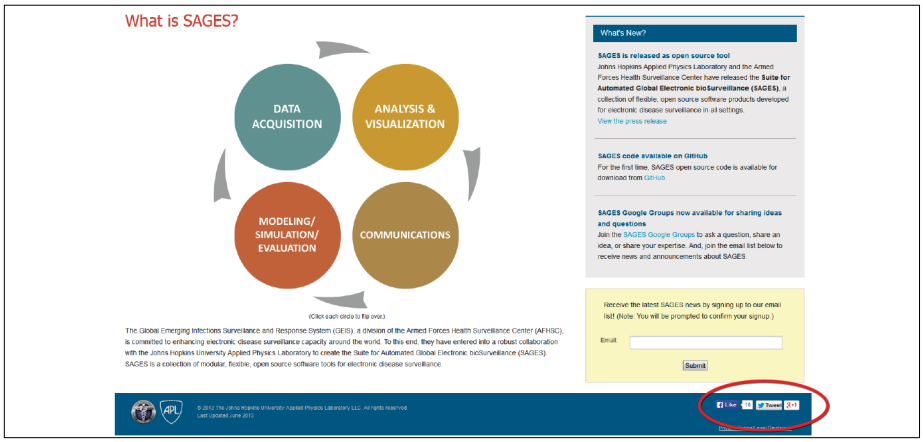

Figure 7. SAGES publicity site integration with Facebook, Twitter, and Google+ social networks (see red circle in bottom right).

DISCUSSION

Two SAGES tools, OpenESSENCE and SAGES Mobile, were publicly released

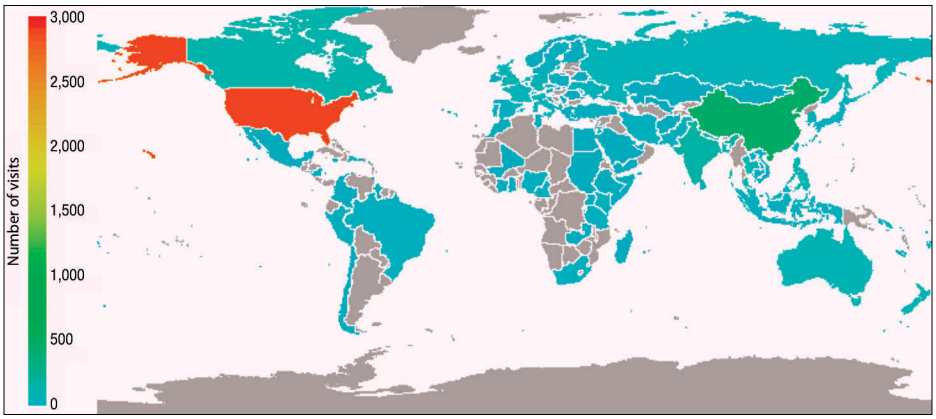

Figure 8. Geographic distribution of visits to the SAGES publicity website between 28 June 2013 and 21 February 2014.

on schedule as open-source software. The open release of source code and binary executable files was complemented by making ample training and informational resources for SAGES available to all current and potential end users. Following in the vein of self-sustainability required for disease surveillance systems to be successful, the open-source hosting and communications infrastructure of SAGES is based on low-cost, sustainable components. In short, this release effort fulfilled the sponsor's direction to "further the development of SAGES as an open-source tool that can be installed and configured without direct U.S. support."[5]

It serves the interests of the USG and the public to foster a vibrant open-source ecosystem by publicly releasing code and related software artifacts, when appropriate, under the terms of an open-source license.[10] Substantial software projects for the public good, such as SAGES, would not be possible if the numerous libraries on which they are built had not been previously released by public and private entities throughout the world. Third-party enhancements to open-source software benefit both the original developers and a wider community of end users, who may even adopt the software as an industry standard and thereby reinforce this virtuous cycle.[6,10] Sponsoring the development of open-source software may even ensure that any organization, and not only the original developers, can be tasked to make future enhancements and modifications to mission-critical software.[13]

There are multiple ways in which SAGES can build on this successful open-source release. Meeting the needs of existing users and potential new users is vital for the continued growth of SAGES, with community considerations being core to several of these needs. In the near term, the SAGES tools, publicity website, and documentation could be translated into additional languages. The collection of SAGES training resources could be expanded to include "how-to" videos posted on YouTube for all segments of the user base, as well as more developer-focused content on the wiki of the SAGES GitHub site. Of course, it will remain critically important to maintain ongoing communications with the end-user community by responding to user inquiries and actively updating the publicity website.

Community considerations will also play a central role over a longer term as SAGES is increasingly adopted. Encouraging existing end users to connect and share knowledge with each other on message forums will help build a searchable repository of technical support information and lessons learned about the SAGES tools. Active end-user involvement will also signal to prospective end users that the software is well supported and widely adopted. Nurturing a base of independent end users and developers will be essential for the longevity and long-term usefulness of SAGES. Eventually, ownership and stewardship of SAGES will transition to the community, perhaps in the form of a nonprofit organization such as the Apache Software Foundation (http://www.apache.org/).

Technical opportunities abound to make the SAGES tools even more useful and accessible to end users. Since the inaugural open-source release, the APL development team has made numerous commits to GitHub that provide incremental improvements to the SAGES tools. As the developer base grows, processes for validating and accepting external code contributions from independent developers will need to be formalized. SAGES itself will benefit by continuing to leverage, and possibly engage with, developments in the wider open-source community. Depending on future demand for EDE, its source code could also be released publicly. Most intriguingly, innovative new tools could be released under the SAGES umbrella. These tools could further extend SAGES' capabilities in public health communications, modeling, simulation, and evaluation of automated surveillance systems.[2] Novel SAGES tools could also take advantage of improvements in underlying technologies to rapidly acquire, process, and report on greater volumes of more-detailed clinical observations and data.[46]

From a policy perspective, the most challenging aspect of this effort was satisfying the multiple policy considerations in a manner consistent with the overall technical, community, and programmatic objectives for the release. We recommend that future open-source release efforts create and update a wiki site that provides a concise summary of policy decisions and opinions related to the release, while also maintaining an audit trail to track all changes made on the wiki.[47] If possible,

it is also suggested that the open-source release effort occur within a short time period after the decision to go open source, so that the same core group of key contributors is available to provide the context necessary for shepherding the software all the way through release.

This effort showcases effective cross-enterprise collaboration in support of a critical sponsor mission and in service of global public health, an objective shared more widely by other divisions of the Johns Hopkins University (see, for example, http://www.hopkinsglobalhealth.org/ and http://www.jhumhealth.org/). Its success provides a guide for existing project teams that may seek to release their work as open-source software. Project teams involved in future efforts for which open-source software is intended to be released should agree on language endorsing that intention when contracts are negotiated.[13] Organizations within the defense establishment have even begun to network among each other to discuss the advantages of using and contributing to the open-source community (see http://www.mil-oss.org/).

CONCLUSION

We have described the successful effort to release existing SAGES tools for international disease surveillance as open-source software. Releasing open-source software is inherently a multifaceted task that involves addressing numerous policy, technical, and community considerations. This effort succeeded through the collaboration of a technically diverse team that benefited from the lessons learned through prior open-source releases of other software. The team was also motivated by the underlying purpose of SAGES to better global public health in a sustainable manner through the innovative application of modern information technologies.

ACKNOWLEDGMENTS: We are indebted to the entire SAGES project team and interns for their critical contributions in making this open-source release successful. We are especially grateful to Shraddha Patel for her close collaboration on the open-source release as well as her review of the initial manuscript. We also thank Aisha Ahmad, Laura Glendenning, Shaku Harshavardhana, and Tom Rossberg for their support leading up to the release, as well as Carol Brueggemeier, William Riggs, and Nigel Tzeng for assisting in the research of this manuscript. Finally, we are grateful to the peer reviewers of this manuscript for offering feedback that helped improve the clarity of the discussion. The views expressed are those of the author and do not reflect the official policy or position of the DoD or the USG.

REFERENCES

[1]Chretien, J. P., Burkom, H. S., Sedyaningsih, E. R., Larasati, R. P., Lescano, A. G., et al., "Syndromic Surveillance: Adapting Innovations to Developing Settings," *PLoS Med.* **5**(3), 367–372 (2008).

[2]Lewis, S. L., Feighner, B. H., Loschen, W. A., Wojcik, R. A., Skora, J. F., et al. "SAGES: A Suite of Freely-Available Software Tools for Electronic Disease Surveillance in Resource-Limited Settings," *PLoS One* **6**(5), e19750, doi: 10.1371/journal.pone.0019750 (2011).

[3]The Johns Hopkins University Applied Physics Laboratory, *Johns Hopkins APL Releases Open-Source Electronic Disease Surveillance Software*, Press Release, http://www.jhuapl.edu/newscenter/pressreleases/2013/130701.asp (1 July 2013).

[4]Campbell, T. C., Hodanics, C. J., Babin, S. M., Poku, A. M., Wojcik, R. A., et al., "Developing Open Source, Self-Contained Disease Surveillance Software Applications for Use in Resource-Limited Settings," *BMC Med. Inform. Decis. Mak.* **12**(99), 1–15 (2012).

[5]Armed Forces Health Surveillance Center, *Capacity Building and Other Efforts*, http://www.afhsc.mil/geisCapBuild (accessed 2 Mar 2014).

[6]Chief Information Officer, U.S. Department of Defense, *Open Source Software (OSS) FAQ*, http://dodcio.defense.gov/OpenSourceSoftwareFAQ.aspx (accessed 13 Jan 2014).

[7]Fogel, K., *Producing Open Source Software: How to Run a Successful Free Software Project*, http://producingoss.com/en/producingoss.html (accessed 3 Sep 2013).

[8]Institute for Dynamic Educational Advancement (IDEA), "Open Source vs. Proprietary Software," *idea* (blog), 22 July 2011, http://www.idea.org/blog/2011/07/22/open-source-vs-proprietary-software/.

[9]Open Source Initiative, *Licenses by Name*, http://opensource.org/licenses/alphabetical (accessed 9 Feb 2014).

[10]Wennergren, D., *Clarifying Guidance Regarding Open Source Software (OSS)*, Memorandum, Department of Defense Chief Information Officer, Washington, DC, http://dodcio.defense.gov/Portals/0/Documents/OSSFAQ/2009OSS.pdf (16 Oct 2009).

[11]National Defense Authorization Act for Fiscal Year 2012, Pub. L. No. 112–81 (31 Dec 2011), http://www.gpo.gov/fdsys/pkg/PLAW-112publ81/pdf/PLAW-112publ81.pdf.

[12]Clark, R., and Payne, B. D., *OpenStack Security Group: Status Update and Plans*, https://region-a.geo-1.objects.hpcloudsvc.com/v1/90187386515277/Public/ossg-apr2013.pdf (accessed 5 Feb 2014).

[13]Saunders, R., and Allen, G., "Cracking the Code: Contracting for Open Source Software," in *Proc. Interservice/Industry Training, Simulation, and Education Conf. (I/ITSEC)*, http://ntsa.metapress.com/link.asp?id=q374n5jg62806751 (2012).

[14]Poku, A. M., *Status SMS Data Collection for SAGES – Leveraging Open Source Solutions to Expand Capabilities*, Technical Memorandum QTH-12-0008, JHU/APL, Laurel, MD (26 Mar 2012).

[15]Gordon, R., "6 Things to Know About Successful (and Failed) Open-Source Software," *Idea Lab* (blog), 1 Aug 2013, http://www.pbs.org/idealab/2013/08/6-things-to-know-about-successful-open-source-software/.

[16]Office of Export Enforcement, Bureau of Industry and Security, U.S. Department of Commerce, *Penalties*, https://www.bis.doc.gov/index.php/enforcement/oee/penalties (accessed 9 Feb 2014).

[17]Feighner, B., *SAGES Open Source Software Release Process*, Technical Memorandum QAI-13-0006, JHU/APL, Laurel, MD (28 June 2013).

[18]Ashar, R. J., *Developer Guidelines for Releasing SAGES Code to GitHub (1.0.0)*, Technical Memorandum QAI-13-0014, JHU/APL, Laurel, MD (30 July 2013).

[19]Otra, comment on Jaynathan Leung, "Bouncycastle Elliptic Curve Encryption on Android," *Stack Overflow*, http://stackoverflow.com/questions/11192686/bouncycastle-elliptic-curve-encryption-on-android (25 June 2012).

[20]Open Source Initiative, *Frequently Answered Questions*, http://opensource.org/faq (accessed 14 Feb 2014).

[21]GNU Operating System, *GNU General Public License, Version 2*, https://www.gnu.org/licenses/old-licenses/gpl-2.0.html (Jun 1991).

[22]GNU Operating System, *GNU General Public License, Version 3*, https://www.gnu.org/copyleft/gpl.html (29 Jun 2007).

[23]The Apache Software Foundation, *Apache License v2.0 and GPL Compatibility*, http://www.apache.org/licenses/GPL-compatibility.html (accessed 15 Feb 2014).

[24]Scacchi, W., Alspaugh, T. A., and Asuncion, H., *Investigating the Acquisition of Software Systems that Rely on Open Architecture and Open Source Software*, Naval Postgraduate School Acquisition Research Sponsored Report Series, Report UCI-AM-10-021, http://acquisitionresearch.org/files/FY2010/UCI-AM-10-021.pdf (Mar 2010).

[25]GitHub, Inc. *Choosing an OSS License Doesn't Need to Be Scary*, http://choosealicense.com/ (accessed 9 Feb 2014).
[26]The Apache Software Foundation, *Apache License, Version 2.0*, http://www.apache.org/licenses/LICENSE-2.0.html (Jan 2004).
[27]The Johns Hopkins University Applied Physics Laboratory, "OpenESSENCE Third-Party Dependencies," *SAGES website*, http://www.jhuapl.edu/sages/downloads/DLFiles/OpenESSENCE_Third-PartyDependencies.csv (accessed 23 Feb 2014).
[28]The Johns Hopkins University Applied Physics Laboratory, "SAGES Mobile Third-Party Dependencies," *SAGES website*, http://www.jhuapl.edu/sages/downloads/DLFiles/SAGESMobile_ThirdPartyDependencies.csv (accessed 23 Feb 2014).
[29]The Johns Hopkins University Applied Physics Laboratory, "EDE Third-Party Dependencies," *SAGES website*, http://www.jhuapl.edu/sages/downloads/DLFiles/EDE_ThirdPartyDependencies.csv (accessed 23 Feb 2014).
[30]The Apache Software Foundation, *ASF Legal Previously Asked Questions*, http://www.apache.org/legal/resolved.html (accessed 23 Feb 2014).
[31]DigitalRoss, comment on crashintoty, "CVS or SVN or GIT?," *Stack Overflow*, http://stackoverflow.com/questions/5545639/cvs-or-svn-or-git (4 Apr 2011).
[32]Wikipedia contributors, "Comparison of Open Source Software Hosting Facilities," *Wikipedia, The Free Encyclopedia*, http://en.wikipedia.org/wiki/Comparison_of_open_source_software_hosting_facilities (accessed 3 Jan 2013).
[33]Harvey, C., "Open Source Software: Top 59 Sites," *Datamation*, http://www.datamation.com/osrc/article.php/3925806/Open-Source-Software-Top-59-Sites.htm (23 Feb 2011).
[34]Gitorious, The Gluon Project, https://gitorious.org/gluon (accessed 28 Jan 2013).
[35]The Apache Software Foundation, *Welcome - Apache Incubator*, http://incubator.apache.org/ (accessed 29 Jan 2013).
[36]Atlassian, *Branch or Fork Your Repository*, https://confluence.atlassian.com/display/BITBUCKET/Branch+or+fork+your+repository (accessed 30 Apr 2014).
[37]GitHub, Inc., *Be Social*, https://help.github.com/articles/be-social (accessed 23 Feb 2014).
[38]GitHub, Inc., *Fork A Repo*, https://help.github.com/articles/fork-a-repo (accessed 23 Feb 2014).
[39]GitHub, Inc., *sages-health*, https://github.com/sages-health (accessed 23 Feb 2014).
[40]GitHub, Inc., *sages-health/openessence*, https://github.com/sages-health/openessence (accessed 23 Feb 2014).
[41]GitHub, Inc., *Introducing GitHub Traffic Analytics*, https://github.com/blog/1672-introducing-github-traffic-analytics (7 Jan 2014).
[42]Google Groups, *sages-health-epi*, https://groups.google.com/forum/#!forum/sages-health-epi (accessed 16 Feb 2014).
[43]Google Groups, *sages-health-support*, https://groups.google.com/forum/#!forum/sages-health-support (accessed 16 Feb 2014).
[44]Google Groups, *sages-health-announce*, https://groups.google.com/forum/#!forum/sages-health-announce (accessed 16 Feb 2014).
[45]Knight, K., "Responsive Web Design: What It Is and How To Use It," *Smashing Magazine*, http://coding.smashingmagazine.com/2011/01/12/guidelines-for-responsive-web-design/ (12 Jan 2011).
[46]Ashar, R., Lewis, S., Blazes, D. L., and Chretien, J. P., "Applying Information and Communications Technologies to Collect Health Data from Remote Settings: A Systematic Assessment of Current Technologies," *J. Biomed. Inform.* **43**(2), 332–341, doi: 10.1016/j.jbi.2009.11.009 (2010).
[47]Wikipedia contributors, "Wiki," *Wikipedia, The Free Encyclopedia*, http://en.wikipedia.org/wiki/Wiki (accessed 2 May 2014).

The Author

Raj J. Ashar is a software engineer in the Bio-Threat Awareness Systems Group in the Asymmetric Operations Sector and a member of the Senior Professional Staff. He led the effort to release SAGES as open-source software on GitHub and has also contributed to other public health informatics, homeland protection, and defense programs at APL since 2003. His e-mail address is raj.ashar@jhuapl.edu.

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