Global Goods Software for the Immunization Cold Chain

Leave Authors Anonymous
for Submission

Leave Authors Anonymous
for Submission

Leave Authors Anonymous
for Submission

ABSTRACT

This paper explores the challenge of taking *Global Goods* software to international scale. *Global Goods* software is non-commercial software designed to support global development goals. We argue that a fundamental challenge behind this type of software is the different roles of the global organizations that fund projects, the country leadership that controls implementation, and the actual users of software. To address this, it necessary to have a design process that balances interests of stakeholders and a technical design that allows for modularity and extensibility. We present a case study of an application for country level management of the immunization cold chain that we have developed and contrast it with major *Global Goods* software systems such as DHIS2 and OpenMRS. The contributions of the work include the design of a pipeline for building a *Global Goods* application that is deployed across multiple countries, a collection of lessons learned during system design and implementation, and a comparison of extensibility strategies of different global goods applications.

CCS Concepts

•Applied computing → **Health care information systems;** *Health informatics;* **•Information systems** → Mobile information processing systems;

1. INTRODUCTION

This paper explores the challenge of taking *Global Goods* software to international scale. We examine a software system developed by the authors as a case study and analyze it in the context of other *Global Goods* software systems that have successfully reached global scale. By *Global Goods* software [24] we mean software designed to support global development objectives which is developed and deployed through non-commercial mechanisms. These systems generally aspire to meet a set of characteristics defined in the *Principles for Digital Development* [4]. Examples of *Global Goods* software include *DHIS2* [8], *OpenMRS*[21], *OpenLMIS*[30], and *Open Data Kit* [10, 14].

Many of these projects have ambitions for global scale to amplify impact as achieving scale can diffuse research and development costs. Additionally, replicating the same software systems in multiple countries increases the efficiency of global organizations ability to support individual countries. However, many *Global Goods* projects only reach the pilot stage. This is a source of critiques of the Information and Communication Technologies for Development (ICTD) field [17]. There are many rationales to conduct small projects and also many justifications for small projects to stay small. However, this paper focuses on identifying software system characteristics needed to achieve global scale.

The topic of scaling software systems (primarily in the health domain) in low-income countries has been an active topic in the health information systems literature. General principles for designing health information systems in low-resource countries have been proposed in multiple publications [18, 19, 22]. The University of Oslo, through the HISP program, has been a thought leader in the process of scaling health information systems with papers describing managing standards [7], local problem solving [27], the politics behind system adoption [28], and field deployment studies [8, 9]. One approach proposed by Mwanyika [23] is to define a *Global Architecture*, which can then be specialized either as a *Global Solution* or a *Country Architecture*, to develop a *Country Solution*. The technical steps to go from an "Architecture" to a "Solution" is one of the key topics of this paper. In addition to works looking at the positive steps towards scaling, it is important to consider the obstacles to taking systems to scale, which is addressed in a barriers literature by authors such as Heeks [15, 16].

A challenge for many global development organizations is finding "appropriate" technology designed to operate effectively in a variety of environments with infrastructure and human resource constraints. To enable scalable software systems, designers need to create abstractions that enable countries to customize and implement applications that have varying requirements such as different languages, data hierarchies, data fields, and input forms. This can be further generalized to organizations that often need to adapt software to the local context requiring the customizability of both the *workflow* and the *dataflow*. In this paper we examine multiple *Global Goods* software systems and report on common features that enable international scale. These include abstractions to enable domain-experts to customize to the local deployment contexts but maintain the system's ability to produce an output that is usable for global data analysis, comparison, and action. This paper describes the design of a *Global Good* information system to support a country's immunization cold chain equipment.

2. IMMUNIZATION COLD CHAIN

Immunization is one of the most impactful public health interventions ever. The near eradication of polio is a testament to the success of vaccines. Polio has gone from being endemic

COMPASS 2020, June 15–17, 2020, Guayaquil, Ecuador

ISBN 978-1-4503-6708-0/20/04. . . $15.00

DOI: https://doi.org/10.1145/3313831.XXXXXXX

in most of the world in 1960, to being endemic in only two or three countries in 2020.[1] Polio cases have dropped by 99.95% since serious eradication efforts began in 1988. Equally significant is the high rate of vaccine coverage for basic childhood diseases such as diphtheria, tetanus, and pertussis, meaning that children do not die from diseases that used to kill them in large numbers.

Immunization has been a global priority since 1974 when the World Health Organization (WHO) announced the Expanded Program on Immunization, which lead to the founding of a group of organizations which play dominant roles in supporting vaccine programs in developing countries. WHO plays a global role in setting policies for immunization, supporting the introduction of new vaccines, and providing technical assistance. UNICEF directly supports countries in vaccine supply and cold chain management. Gavi, the Vaccine Alliance is primarily a funder of vaccines, providing vaccines for free or low cost to eligible countries. These organizations are funded by government international development programs and large private donors. These global organizations have a very big influence on country immunization programs both by setting policies and by providing much of the funding for vaccines distribution and logistics. A significant fraction of vaccine refrigerators are supplied to countries directly by donors. A list of officially sanctioned equipment, the PQS list, is maintained by WHO, which specifies eligible equipment. At the global level, there are complementary activities of providing oversight to countries as well as determining allocation of equipment to countries based on need and budget.

Central to immunization is the logistics system that distributes vaccines from global manufacturers to the point of use [32]. Cold chain equipment is generally managed centrally with countries having a hierarchy of vaccine storage facilities. The hierarchy usually has a national vaccine warehouse that supplies regional storage facilities which then distribute vaccines to hospitals, health centers and clinics. Vaccines need to be kept cool so that they maintain their potency, so cold storage is a critical component of the supply chain. Specially made vaccine refrigerators are designed to keep temperatures in a specific range (2C to 8C), be robust to power failures, and use different fuel sources. Country-level equipment management generally includes maintenance and repair of equipment, as well as allocation of new equipment. Countries with strong cold chain management will also have plans for retirement and replacement of older equipment. An example of long range planning for the cold chain was the phase out of chlorofluorocarbon refrigerators in the 1990's. A current challenge is the replacement all kerosene and gas refrigerators because of fuel costs and fire danger. There is a goal of expanding the use of solar power for vaccine refrigerators through solar direct drive refrigerators that do not require batteries.

Country management of cold chain equipment is highly dependent on having an accurate understanding of equipment at different facilities. Unfortunately, very few low-resource countries keep an accurate and up to date inventory of cold chain equipment. A standard approach has been to conduct county wide cold chain assessments at high cost and then let this information get out of date. Even countries that make an effort to maintain centralized data have questions about the accuracy of their data. The real challenge is visibility and creating a consistent view of information. Countries do have management structures in place, so that there are usually supervisors available a the district level who have responsibility for vaccine logistics including managing the cold chain. The problem is that the district information often does not align with what is known at the national level. For example, we have encountered the situation where the health facilities that the district supervisors are responsible differ from the health facilities recorded at the national level.

Many of the challenges faced by cold chain equipment managers could be made easier with accurate information. Our goal is to create a system that helps countries ensure a high quality vaccine cold chain exists which has sufficient working vaccine storage space to keep vaccines at appropriate temperatures until they are delivered to beneficiaries. This paper summarizes our efforts to create a globally reusable information system to support accurate management of country wide immunization cold chain equipment. Countries could improve vaccine delivery with up to date information on which vaccine refrigerators are working and which are broken and needing repair. Global organizations need better information to conduct long-term planning to support a country's cold chain. Information about which models of refrigerators have the lowest rate of failures in field contexts could be easily determined if equipment data was available in a format that allowed for direct comparison.

3. GLOBAL GOODS SOFTWARE

The term *Global Goods* is used to indicate that these software products are being made available for world wide access to create a digital infrastructure for global development, with a secondary meaning that there is public benefit associated with these systems. *Global Goods* software refers to software products that support global development goals. Thus many international development organizations integrate *Global Goods* software in their interventions in attempt to obtain technology efficiency gains observed in other contexts. The aim is to leverage technology to improve process and decision making with an expectation that these improvements will lead to gains in global development outcomes. Global development funding agencies are also recommending that organizations apply evidence-based development which requires gathering of data to make decisions. This means that information systems are becoming indispensable for organizations to make informed decisions as digitizing both the *workflow* and the *dataflow* often improves data accuracy, improves the timeliness of information, increases accountability, and increases data visibility to decision makers. Having reliable data also helps decision makers with monitoring and evaluation, selecting where services should be delivered, planning and managing resources, and comparing various interventions' results.

[1] Wild polio cases are still found in Pakistan and Afghanistan. Polio is classified as still being endemic in Nigeria as the security situation makes it difficult to fully assess transmission status.

Most *Global Goods* projects are motivated by the desire to have a positive impact and advance international development goals. Generally *Global Goods* software are made available at little or no cost to countries for local deployment. Specifically, we define *Global Goods* as reusable software frameworks that operate with a free and open-source software (FOSS) model allowing any organization to obtain the software for free but their could be costs to operate the software such as low cost fees for cloud-hosting or Internet connectivity. These systems are generally supported by communities of developers and implementers with various funding models (e.g., grant-based, consulting fees, hosting fees). There is also a recognition that these systems should strive towards compatibility.

3.1 Example Systems

There are a number of systems that can be considered as *Global Goods* software. We focus on several systems that have achieved scale with deployment in dozens of countries at a scale. There are other projects for civil registration (e.g., OpenCRVS), community health (e.g., CommCare[13]), health insurance (e.g., openIMIS), and human resource management (e.g., iHRIS) that are also *Global Goods* software as well. However, because of space limitations we chose representative systems from different categories that are established *Global Goods*.

DHIS2 District Health Information Software 2 (DHIS2) is an web-based Health Management Information System [8]. A core functionality provided by DHIS2 reporting of health indicators from facilities and local regions to national regions. DHIS2 has been adopted by many countries as their national health reporting tool. DHIS2 is managed by the Health Information Systems Program at the University of Oslo in Norway.

OpenMRS – Open Medical Record System (OpenMRS) is a widely used, open source platform for implementing medical record systems[21]. The system got its start supporting HIV/AIDS care and treatment and has grown to support primary health care in a broad range of settings. OpenMRS allows custom data structures but uses a common-concept dictionary to link vocabulary and reporting. OpenMRS also supports a software-plugin framework making it easy to add custom modules to adapt to specific use cases. An estimated 3,000 sites now deploy OpenMRS.

OpenLMIS – Open Logistics Management Information System is a cloud based system designed to manage health commodity supply chains in developing countries. The system supports inventory management, ordering and fulfillment and includes reporting and analytics. VillageReach, an NGO, is the leading contributor to OpenLMIS and manages the project as an open source project[30].

Open Data Kit versions 1 & 2 – Open Data Kit (ODK) [10, 14] was created to empower resource-constrained organizations to build information services in low-resource contexts. ODK simplifies the process of building and scaling an information management solutions by providing two tool suites that 1) are customizable by a non-programmer and 2) can operate in disconnected environments. The first ODK tool suite called "ODK 1" [14] was designed to enhance and replace paper data collection. It used XForms to specify the data collection in a unidirectional data flow (similar to paper). Many organizations requested new features that could support more complex *dataflows* and *workflows* that included functionality to synchronize data back to mobile clients for field workers to review and update [11]. This feedback led to a second ODK tool suite called "ODK 2" that focuses on bidirectional "data management" instead of unidirectional "data collection" [10]. (In 2019, ODK 2 was renamed ODK-X to emphasize that it is a different tool suite from the original ODK 1 and we refer to it as ODK-X for the remainder of the paper.

99 DOTS – 99DOTS was created by Microsoft Research India to perform low-cost monitoring of patients taking Tuberculosis medication [12]. Patients send a free call each time they take their medication enabling health providers to monitor the patients adherence to the medication dosage schedule. 99DOTS works by packaging custom secondary envelopes around patient medication with a series of hidden numbers behind the pills. Patients report taking the medication by contacting a toll-free number that was hidden behind the pill. 99DOTS relies on most patients having access to a mobile phone capable of making a toll-free call. The reliance on voice dialing and not the Internet enabled the system to reach the maximum number of patients. Health workers responsible for monitoring medication adherence can view a patient's medication history via a mobile app, a website, or SMS messages.

Ushahadi – Ushahidi, which in Swahili means "testimony", is a crisis-mapping software project developed during the 2008 Kenyan elections to enable people to communicate issues and document post-election violence. Ushahidi continued as an open-source project that makes it easy to crowd-source information so that everyone's voice is heard [6]. Ushahidi is designed to enable individuals or groups to both collect and disseminate information to and from people in the field.

FrontlineSMS – FrontlineSMS [1] enables organizations to send SMS messages at scale. For collecting small pieces of data over SMS can simplify crowd-sourcing data, community communication, and collecting research data. Features such as automated responses, triggered prompts, and keyword search make FrontlineSMS easy to adapt to many use cases.

3.2 Common Features of Global Good Software

All the *Global Good* software frameworks we examined focus on gathering accurate information and aggregating the data into a usable format for a specific organization. Additionally, we observed that the systems are designed to enable organizations to adapt the software frameworks to the local context requiring the customizability of both the *workflow* and the *dataflow*. We examined multiple global good software systems and found the following common features: 1) customizable to deployment context, 2) modular design, and 3) data update refresh requirements are often long, in terms of hours or days instead of seconds. A comparison of *Global Goods* software design is shown in Table 1.

One design decision that a *Global Goods* software project will make is whether to be a subject domain-dependent system or subject domain-independent system. There are advantages to

Table 1. Global Goods Software Frameworks Common Design Feature Comparison

	DHIS2	*OpenMRS*	*OpenLMIS*	*ODK 1*	*ODK 2*	*99 Dots*	*Ushahidi*	*FrontlineSMS*
Scalable	X	X	X	X	X	X	X	X
Modular Design	X	X	X	X	X			
Localization	X	X	X	X	X	X	X	X
Abstractions create limitations	X	X	X	X	X	X	X	X
Subject domain independent				X	X		X	X
Subject domain dependent	X	X	X			X		
Flexible data structures	X	X	X	X	X			
Disconnected operation		X		X	X			
Health Workers receive data via Internet	X	X	X	X	X	X	X	
Patients communicate via Voice or SMS						X	X	X
Data update/refresh longer than seconds	X	X	X	X	X	X	X	X
Supports predefined roles beyond data collector and system administrator	X	X	X		X	X	X	
Customization abstractions designed for non-programmers	X	X	X	X	X	X	X	X

either approach as a domain-independent can be used more broadly by many organizations working in different domains. However, a domain-dependent solution can make it easier for organizations to use by 1) shrinking the number of options that need to be configured for the system to be used 2) minimizing the vocabulary being used to the domain space so that system features are more intuitive to users, and 3) having a set of predefined roles (e.g., patients, health care workers, supervisors) beyond that of data collector and system administrator that enable customized interfaces that display appropriate information to a user's role. However, the scope of the domain-dependence can also vary. For example, both OpenLMIS and DHIS2 are both information managements for the 'health' domain, but OpenLMIS is a logistics management information system while DHIS2 is a health management information system. A logistics information management system focuses on the amount of medicine, vaccines, and other supplies are at each facility and assists with ordering and allocating new supplies. Whereas a health management information system tracks incidence of disease, health services rendered, and patient information. Another common design feature is that health care workers often need more rich data than can be provided by voice or SMS so most of the systems have the "field worker" or "health care" worker send and receive information via the Internet.

3.3 Fitting within Existing Eco-systems

While global goods projects have generally been led by technologists, there is a recognition that the technologies used must be suitable for the deployment context. A global good software system has to adapt to the existing eco-systems that exist in international development. The majority of the global goods mentioned above have been active projects with much evolution in scope and technology. For example, DHIS2 is the outgrowth of work of activists in post-apartheid South Africa [8] who were working to make the health system more equitable. Early versions of DHIS (the predecessor to DHIS2) had failures in several countries, including Cuba [26] before a number of implementations took off and reached national scale. Both OpenMRS and OpenLMIS have histories with varied types of deployments and significant redesign of their products. The length of time these projects have been active has been important for building expertise and establishing themselves in the domain and donor networks.

Software development eco-system

Global goods systems are generally developed with the goal of achieving social impact and not for commercial reasons. Many of these systems have an academic pedigree: DHIS2 from University of Oslo, OpenMRS is from University of Indiana, and CommCare is an outgrowth of projects from Harvard. The remaining systems developed by NGOs, either to support internal projects, such as Village Reach developing VRLmis, the predecessor of OpenLMIS, for its internal projects, or developed under the direction of a donor, such as iHRIS, which was developed by IntraHealth under USAID contracts. The supporting ecosystem has passionate, talented, software developers, but in many cases, they work with less stability than is provided by the software industry, either as students, open source developers, or as contractors.

Funding eco-system

As global goods software systems are non-commercial, some mechanism is needed to fund the cost of software development and system deployment. Even though the software is free there are still costs associated with using a FOSS systems. There are costs associated with purchasing computing hardware, as well as upkeep costs of keeping the computers, servers, and mobile devices operating. Additionally, there is the time needed to configure the FOSS system's *dataflows* and *workflows* so it is usable within the deployment context. There are also costs associated with training the people who use the software. Furthermore, technology is not static, so as the global technology eco-system evolves, the FOSS system also needs to continue to upgrade to the latest technology releases. To keep FOSS systems operational, someone needs to constantly update the system with eco-system changes such as operating systems API changes, library bug fixes, and security fixes.

Most of the global goods projects rely on donor and grant funding for core software development activities. This leads to a fairly complex web of funding for systems, including a mix of funding for core development and deployments. Funding organizations often have a very big influence on what features are implemented and what types of behavior will be supported. As donor funding is often target to particular health domains, certain use cases are targeted based on availability of funding. The developers of global goods projects often identify software maintenance as a particularly difficult area to fund.

Deployment eco-system

There is a recognition that the technologies used must be suitable for the deployment context. There are standard localization issues (such as language and display formats), along with more complicated issues of adapting applications to local contexts and *workflows*. This may be accomplished by having a customization layer (such as the entity/attribute approach of DHIS2) or supporting inclusion of custom modules as in OpenMRS. Global goods projects are often very concerned about the code-base diverging for different countries, and ensuring that county instances are based on a current version of the core software.

The evolution of the DHIS project is a good example of different *dataflows*. DHIS was initially an Microsoft Access application which relied on *Feed Forward* files for sharing data, and was appropriate for non-networked PCs of the time. DHIS2 made the shift to a Java based system. Ushahidi's success comes from the recognition in 2008 that in Kenya that the SMS channel of the mobile phone was the best platform to use for crowd-sourcing.

Deploying health related software systems at scale means integrating them into the government health system. In most countries there is a Ministry of Health (MoH) that is organized around a set of verticals such as infectious diseases, maternal and child health, and immunization. While the administrative hierarchy may have a number of levels, the district level is often the operational level for health system functions. For example, reporting is centralized from the district levels, and there will be managers for programs such as malaria, tuberculous, and immunization. Requirements from donors often lead to standards being imposed on countries. For example, electronic medical record systems, such as OpenMRS, grew out of funders trying to improve clinical support of HIV treatment. DHIS2 similarly grew based on supporting US PEPFAR reporting requirements as well as countries needs to report across domains. Countries are taking a greater level of control of their programs, which is generally in line with the goals of donors. This means that adoption of systems is dependent on government support and there are considerations such as local management of systems and control of data which are important. Technical capacity in country is increasing, so these trends will help systems become sustainable.

4. DESIGN OF A COLD CHAIN INFORMATION SYSTEM

Collecting and disseminating accurate data regarding a country's cold chain improves resource-allocation and planning. Unfortunately, many cold chain equipment inventories are paper-based systems that contain large amounts of inaccurate or out-of-date information. To address this, a software system is needed that is capable of supporting a country-wide cold chain inventory update that includes visits of remote facilities so the status of the cold chain equipment can be accurately determined. The goal is to have a updated database showing the current status of every vaccine refrigerator in a country so that it is possible to identify where there might be problems in the cold chain (broken equipment or inadequate capacity to store vaccines), as well as to support all needed reporting and planning activities. By using a common global application, country level information should be able to be easily aggregated to support global analysis.

Replacing the paper-based system with a mobile tool such as ODK-X can improve the speed and reliability of the inventory update process. Remote field workers can download the most up-to-date cold chain data to their mobile device and visit sites while disconnected from the network. The worker uses ODK-X to enter updates or new refrigerator information. Once Internet connection is available, the updated cold chain data can be synchronized to the cloud and made immediately available to decision-makers. A software system that supports a remote workforce making real-time updates on location will significantly improve the inventory update process and decrease the duplication of work between fieldworkers because a synchronized mobile device will have current information. The tie between cold chain monitoring and ODK-X is standing, with a cold chain application proposed in the original vision paper explaining the rationale to create ODK-X [11].

One goal of the immunization cold chain project is to design a reusable *Global Goods* application that can be deployed in at least 70 countries.[2] The application is designed to support a set of cold chain management tasks that were refined by years of experience working in the domain. These tasks were determined by extensive discussions with country and global stakeholders so that successfully incorporating these tasks is the value proposition for the application.

- Inventory of cold chain equipment. Update the national cold chain inventory by supporting collection of information from health facilities.
- Routine reporting and updates. Regular reporting of the status of cold chain equipment and refrigerator temperatures. This is commonly done on paper.
- Surveillance. Regularly tracking of equipment performance. This can be important for studying the deployment of new types of vaccine refrigerators such as solar direct drive refrigerators.
- Equipment management. Track repairs of equipment and allow the reporting of equipment needing repair.
- Country management of the cold chain. Provide overview information on the status of the cold chain to inform decision makers and allow compilation of reports for the global level.

4.1 Mobile Application

For the cold chain system, an appropriate technology is Android phones using the mobile network for data, and cloud hosting for the back end. Android phones are no longer a novelty in developing countries, and many workers have there own Android phones. Additionally, Android is the most common smartphone OS with more than an 73% market share as of February 2020 [3]. There is even push-back from some workers in being given Android smartphones for the project as they would rather run the app on their personal Android

[2] The target is all of the countries receiving GAVI support for their vaccine programs.

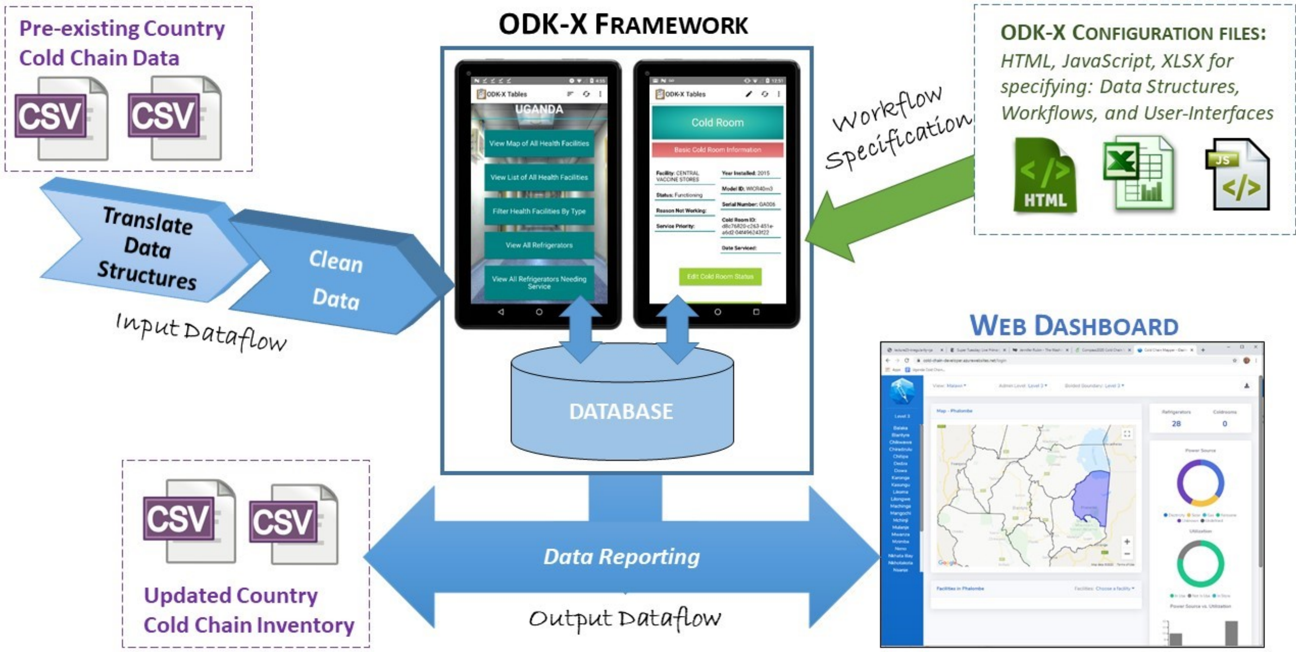

Figure 1. An architecture diagram of the immunization cold chain application that shows how the *dataflows* through the various pieces of software.

devices. Data costs are now low enough that it can be born as a personal expense.

ODK-X [10] is designed to operate on Android compatible devices in economically constrained environments. ODK-X's customizable workflow, user permissions, rule and adherence enforcement, disconnected operations, and eventual data synchronization were critical features to coordinate country-wide cold chain inventory updates. In the App, the main navigation was to health facilities which was done through a geographic hierarchy or through a map based interface if health facility coordinates were available. From the health facility pages, the user can access the lists of available refrigerators to update the information. Additional information is available to the user about the refrigerators including maintenance records and model information. While ODK-X provides a rich permission model with table-level and row-level permissions, the current application provides three permission levels for users: one for the cold chain technicians, one for their supervisors, and one for system administrators. The difference between the permission levels is that some operations (such as adding or deleting a health facility) is only available to their supervisors. ODK-X also provides the option of giving geographically based permissions, but they are not currently being used by the implementing countries.

ODK-X's synchronization protocol is based on a REST architecture that keeps the data on multiple devices synchronized to a master copy stored in the cloud. All synchronization operations are idempotent, thereby simplifying the API because clients do not have to worry about losing data, as requests can be safely repeated in environments where network timeouts occur. To minimize data updates that conflict, data updates are processed as row-based changes to keep changes small. For example, when performing a cold chain inventory, if updates were at a coarse granularity, such as table-based or file-based, a conflict might be detected for two workers updating refrigerators while working at different sites. By keeping conflict detection at the row-level, multiple users can make updates to shared data tables and the system will detect that there is not a conflict as long as the same row not is updated by different users between their synchronizations. Cell-based conflicts would be an even smaller unit of data that would further reduce conflicts; however, in a single row the cell values are often inter-related. We felt that too much context could be lost by treating the cells separately, thus leading to reconciliation errors. In an application such as the cold chain application, conflicts should be rare, as the technicians are working in geographically distinct areas and likely will not update the same rows of the tables within days. Training technicians to synchronize data frequently also helps reduce conflicts. If a conflict does occur, the ODK-X synchronization protocol is designed to have the user determine how best to resolve conflicts. A conflict is defined as two users with different updates to the same row. ODK-X uses table locks on the server to ensure only a single change to a data row can occur at anytime. When the 'runner-up' client finally obtains the lock and attempts to alter the same row, the update be rejected as a conflict. Once a conflict is detected, the user manually determines which version of data is correct between their pending changes on the local client and the updated data row on the server. The rationale for having the user who caused the conflict also resolve the conflict is that the user was recently working with data and is likely to have the necessary information and context on how best to resolve the conflict.

4.2 Cold Chain Data Use Tool

Another component of this system is a tool for visualizing global cold chain inventory data that aims to facilitate better understanding of data at many levels in the cold chain. At the global level, this tool provides stakeholders with the opportunity to efficiently identify areas in need of updated vaccine refrigerator equipment. At a national level, it can provide insights to assist in the distribution of vaccines so that health facilities have sufficient amounts of vaccines for the served populations. The visualizations are based on extensive feedback from relevant stakeholders in order to make sure that they are addressing relevant questions. In addition to providing visualizations, the tool also is the mechanism for exporting the ODK-X database. Discussions with country stakeholders emphasized how important it is to get access to the raw data, and that CSV export is the preferred mechanism so that data can be analyzed with standard tools such as Excel. We export the basic ODK-X tables, and also provide capabilities to provide joins of tables and selection of columns. Some of the internal ODX-X columns, such as last update time are also of interest.

Requirements

In order to create unified visualizations for multiple countries, this tool requires a standardized data schema of the cold chain equipment. For this reason, the visualization tool is based on the ODK-X Cold Chain Application data schema. Another important requirement is the availability of accurate shapefiles that map appropriately to a given country's administrative hierarchy. This allows us to make the visualizations interactive across different levels in the cold chain. Another request was access to temperature monitoring devices that measure refrigerator temperature. Visualizations based on this data are very useful to certain stakeholders. Lastly, the tool is hosted as a web application and thus requires basic internet infrastructure to operate.

Implementation Approach

The dashboard is comprised of an interactive map used to spatially display administrative region borders and allow selection of regions. Administrative levels can be changed to alter the scope of the data exploration from country to district levels. Adjacent to the map are data visualizations of types of power sources and utilization status of refrigerators within the selected region. Below the map, facilities within the region can be selected to determine details of a particular facility, such as facility ID, population served, and refrigerator models.

The front-end of the dashboard is implemented using Embedded JavaScript (EJS) to manage HTML and CSS code. The visualizations are constructed using the Google Maps API and use D3.js, a JavaScript visualization library. The back-end of the dashboard is implemented using Node.js to make API calls and allow data retrieval from our Azure database. The back-end also synchronizes with the ODK-X database everyday to get the most up to date information for visualization.

Data Ownership Issues

An immediate issue involved in the global visualization tool is data ownership. Many countries classify health system data as sensitive, so there is concern with unauthorized personnel accessing data. For this reason, we are deploying separate versions of the tool for each country we are working with. In future versions, we plan to expand to several countries in order to promote insights on an international level. To solve this issue each country will have its own database for the full country inventory that is connected to an ODK-X server for updates. Some of the data is from public sources (such as administrative hierarchies and facility lists) and can be contributed to a central global database. Other information, such as information on refrigerators can be aggregated by region for global reporting. The most detailed information, which is useful for country management, turns out not to be of interest higher up. An area that is important is various aggregate or sample reports that can be used as part of standard immunization program evaluations. For this, the tool exports can be made to align with the reporting requirements, which is a feature requested by both global stakeholder and country program representatives.

4.3 The Data Pipeline

The cold chain application is datacentric. It depends on having an accurate representation of a countries health system hierarchy, and the central use case for the mobile application is managing a cold chain equipment inventory. As each country has it's own database, each country must have it's own instance in a health system hierarchy. From the beginning, we recognized that creating the data pipeline for initializing the database would be a central challenge in building an application for the global immunization system.

Data pipeline

The data pipeline is the pre-processing phase to initialize the application with country specific data. This consists of administrative data (Admin hierarchy, health facility list), geographic data (shape files, GPS coordinates), and cold chain inventory data (refrigerator inventory, refrigerator status). The administrative data is a prerequisite for building the application and structuring the database. Geographic data is needed for some features (such as navigating by facility on a map) and is important for visualization, but the application is functional without it. For initializing the application, the cold chain inventory data is optional, as there are cases where countries would use the tool initiate collection of a cold chain inventory, and other cases where a country has a full inventory to upload into the tool. The overall process is to collect country information, then clean and merge the information into a standard form which is represented as a CSV file. A series of scripts are then used to convert the data into the forms used by ODK-X and the database. The data represented both in a database as well as in ODK-X configuration files. At this stage, scripts are used to perform integrity checks on the data.

Initialization problem

The first step in building a country instance of the the application is to identify the administrative hierarchy and a list of health facilities. As a global application, this is a task that we have taken on for our deployment countries, as opposed to passing it to country implementers. The administrative hierarchy is used both to group facilities in the mobile app as well as for aggregation of data for visualization and analysis. As we discuss in Section 5, there are a surprising number of challenges in working with administrative hierarchies do to inconsistent identification of administrative levels, and changes in administrative units due to splitting or merging of regions. Countries are encouraged to develop and maintain a national master health facility list containing information on all health facilities within the country [29]. This would be ideal to initialize the application with. Unfortunately, many master health

Table 2. Application data sources

Type	Data	Source	Common Challenges
Country	Admin hierarchy	Public repositories	Inconsistent levels, added units, naming
	Facility list	Master facility list, websites	Name resolution, lack of facility code
Geographic	Shapefiles	Public repositories	Matching with admin hierarchy, reducing size
	GPS coordinates	Public lists, maps, app collection	Missing coordinates, erroneous location
Inventory	Refrigerator	Pre-existing inventory, site visits	Matching names to PQS database
	Status	Existing databases, app collected	Inconsistent terminology and different field options

facility lists are incomplete, contain duplicates facilities, or are outdated.

Our approach to building the initial data set of health facilities has often involved identifying multiple lists of health health facilities, and then merging the lists to identify the key fields. This requires addressing the *name resolution problem*. In different health facility datasets, it is common for the same facility to be have multiple different spellings, sometimes due to differences in transliteration from the local language to English or French [25]. Another common challenge is that some versions of a facility name will have a facility type suffix, such as "Hospital" or "Health Center." Another data challenge that arises is that we want to associate each facility with its lowest level in the administrative hierarchy, which may be difficult to find. One potential resolution to this is to match its GPS coordinates (if available) with administrative boundaries from shape files (if available).

Data cleaning

The data used by our system needs to be clean to provide accurate visualizations and analysis. Manual data cleaning is done during the initialization phase. Data entered from the ODK-X app does not need to be cleaned as the app provides input checks. The big challenge in initializing the cold chain equipment inventory is in matching the model of the refrigerators which can be provided as a text string. The problem is greatly simplified by the PQS list issued by WHO of approved refrigerator types. It turns out to be sufficient to identify models outside of the PQS list as "domestic" or "generic." Once refrigerators are identified with a known model, then information about the refrigerator (such as capacity or energy type) can be extracted from a reference table. We expect some of the data cleaning to happen after deployment. For example, inventory data should be checked by site visits and updated accordingly. GPS data can be inaccurate, and it is common to see errors such as health facilities placed in the middle of the ocean in the initial data. These can be updated by the app using the GPS on an Android device.

Shapefiles and tfe Google Map API

To assist geographical analysis, the visualization tool requires accurate shapefiles to be matched with the GPS coordinates of health facilities and the administrative regions. We acquire the shapefiles which are suitable for database or ArcGIS linkage to the targeted administrative hierarchy level from various sources. We managed to acquire most recent boundary polygon and line shapefiles and KMZ files from various GIS data source providers, such as the Humanitarian Data Exchange (HDX) [5], OpenStreetMap, the GADM Archive [2], and the World Bank. Then, we reduce size of the shapefiles by removing redundant points and convert the shapefile to GeoJSON format so it can be layered on a Google map. The Google Maps API creates a base map on which arbitrary geospatial data can be layered. Then, we match the GPS coordinates of health facilities on the Google map and display the geopoints on top of the administrative boundaries shapefile layer.

5. CHALLENGES

In doing this work, and in earlier, related projects we have had extensive conversations with different stakeholders. For the deployments underway, we have had multiple country visits for requirements gathering and feedback. Based on this, there are two important types of challenges to highlight: the difficulties associated with collecting country data for the application, and management of conflicting requirements arising from the global perspective, country management, and the workers who will eventually use the system.

5.1 Data Challenges

This is the type of project which can be summarized as: "The application is the data.," since the goal is to enable management of data for a particular system. The application is structured with *underlying data*, the country administrative structure and health facilities, and the data that is collected on top of it. The fundamental challenge is developing the underlying data to bootstrap the data collection and management activities.

Health Facilities

There is a body of work advocating that countries build master facility lists and keep them up to date [31, 29], but this is more aspirational than reality. Definitions of the minimal amount of information in a master facility list differ, but it usually includes the name, ownership, services offered, and location of the facility. A national master list of health facilities, along with official health facility identification numbers would have greatly simplified our work. There are obviously many difficulties in creating and maintaining this type of list. Our experience has been that there are often multiple facility lists circulating in the ministry of health, managed by different departments with some differences. One issue in building facility lists is whether they just include public facilities, or if they include private health facilities as well. Tracking private health facilities in countries can be very hard with the opening, closing, and moving of different businesses.

Health Facility Name Resolution Problem

A huge challenge in our database initialization is the health facility name resolution problem which arises when attempting to merge health facility lists. Differences in spellings arise from many different sources including differences in transliteration, inconsistent abbreviation, local differences, and random

Table 3. Country Facilities Total Match vs Public Match

Country	Total Matched	Public Matched
Benin	78.09%	80.67%
Malawi	52.60%	56.95%
Kenya	56.74%	66.95%
Uganda	87.46%	84.73%
Zimbabwe	37.63%	40.26%

errors. We have developed multiple tools for name resolution based on heuristics such as name normalization and string edit distance which identify a large number of correspondences which can be identified manually.

The first open-access comprehensive, geo-referenced public health facility database for sub-Saharan Africa was published in July 2019 [20]. It was created by compiling national master health facility lists from a variety of government and non-government sources from 50 countries and islands in sub-Saharan Africa and multiple geocoding methods. It contains a spatial inventory of 98,745 public health facilities. We analyzed this sub-Saharan Africa dataset in comparison to health facility data of several countries we collected through the Cold Chain Equipment Management System (CCEM). A notable difference is that our collected data contains both private and public facilities, while only public facilities were collected in the comprehensive sub-Saharan Africa dataset. The results are shown in Table 3, with each entry being the percent of facilities matched, ignoring case, out of the CCEM dataset. The Total Matched column is using all facilities in the CCEM, while the Public Matched has filtered out all private facilities in the CCEM dataset. Overall these results seem strong considering the differences in spellings and suffixes not accounted for in the matching. While most matched percentages increased on filtering out private facilities, Uganda's decreased, likely due to some discrepancy in recording facility type, showing another potential source of error in data collection.

National Administrative Hierarchies

One of the painful surprises for us was the challenges associated with managing national administrative hierarchies. Countries have different mechanisms for naming administrative levels and not all levels are used consistently. For example, many countries will have analogs of provinces and districts and then an intermediate level such as a division which might or might not be relevant. Handling multiple levels of an administrative hierarchy with different names for levels is something a global application must handle. There are also issues arising in naming of units as discussed above. The biggest challenge is that administrative hierarchies are continually changing. For example, Wikipedia indicates that Pakistan added or removed districts in 2001, 2004, 2005, 2011, 2013 and 2018. The gives the requirement that the application must be robust to changes in the administrative hierarchy, and leads to practical difficulties of tracking administrative regions. The attempted solution of keeping administrative hierarchies matching the current one is complicated by the need to interact with systems or data sources that have out of date hierarchies.

Geographical Data

The availability of accurate shapefiles is important for map visualizations in global, country and local scales. While shapefiles are available on the Internet, the files may not be updated with the most recent geopolitical regions of respective countries. Another problem from the shapefile inconsistencies is unsuccessful matching to the respective country's administrative hierarchy as well as inconsistent spelling of regions. This makes the cross-checking process across provided shapefiles, health facilities data and official geopolitical information challenging. There have been a few countries, such as Democratic Republic of Congo, where we have not been able to find shape files that match the administrative units we were provided. In this case, a centralized and credible GIS data source with the most up-to-date shapefiles and administrative hierarchy would be important to initialize our visualization tool.

5.2 Design challenges

The biggest challenge in design for this application was balancing the needs and requests of different stakeholders. While there was a common vision for the application, and no significant controversy on overall approach or architecture, there were differences on the definitions of particular features for the system. For this application to be successful, it must align with the global stakeholders (who will be primary funders of the work), along with the health ministries of specific countries who will manage the deployment. We now discuss the various factors where country and global interests diverge. The challenge that needs to be addressed is managing conflicting requirements to allow customization per country yet maintaining a common application and code base.

Integration with country information systems

One concern for countries is that they should have a small, cohesive set of information systems, so there is the natural question of how this proposed system would fit with other information systems. On the architectural level, there is the question as to whether the system could be a component of other systems, or if there would be some form of data exchange between systems.

We believe that it is desirable to tightly integrate the cold chain information system with other country information systems; the challenge is that there are a wide number of configurations used by countries, and it would become a programming intensive approach to integrate with each particular system, and would also lead to on-going software maintenance challenges. Our approach at this stage is to propose a *data bridge* for connection between systems, where data is imported and exported in tables, and that common keys and fields can be used for linkages. We note that it is not time critical to update other national systems for the cold chain information system, so that periodic (e.g., daily) batch updates are sufficient.

Scoping the application

Limiting the features of a system such as the cold chain system is a challenge as software is malleable and extensible. The questions "Can the software be made to do X?" is almost always yes, as with sufficient resources, any new features can be added. However, software design principles promote applications having a cohesive logic. For our application, implementing a stakeholder's suggested feature can contradict the goal of building a global application. However, being responsive to stakeholders is important in order to have the system adopted. We illustrate this with two specific examples.

One example is adding a capability for vaccine stock manage, where the amount of vaccine stored at a facility is reported on a monthly basis. A basic version of this would be very easy to add to the system as an additional ODK-X form to report vaccine stock levels, and the vaccine stock reporting could be done by the same user who is managing the cold chain. However, the risk is that this expands the application to managing logistics and would suggest a growing list of features including reporting stock outs, computing stock targets, and eventually managing ordering.

Another suggestion was the automatic uploading of data from temperature recording devices placed in refrigerators. A worker is required to do a monthly report based on the device, so instead of manually extracting data, a richer data set could be directly uploaded from the device by USB to an Android phone. This suggested feature fits better with the overall logic of the application, and presents less of a risk of moving into a new class of application. The upload of temperature data also aligns with the global stakeholders who are actively supporting this type of monitoring.

Country vs Global Application

The goal of the project is to design an application that can be deployed in at least 70 countries. There are standard localization issues, such as language and display formats, that must be addressed, and Section 5.1 discusses challenges associated with different configurations of administrative hierarchies. One issue that comes up frequently is different reporting standards for particular fields of data. To give a very specific example, what are the different ways to capture the working status of a refrigerator: Is it a binary decision of working versus not-working, or is there and intermediate working-needs-servicing status? This particular question has led to long discussions with no resolution. Another area that has emerged as controversial is the classification of types of refrigerator failures. If the application were deployed in a single country, it would be easy to implement particular country choices, but the difficulty is aligning these across different countries. For cases where a standard has been established, possibly through some widely adopted tool, implementing to a standard solves this problems. There are additional challenges that come up in adapting the application to country workflows such as determining who has permission for certain operations.

6. DISCUSSION AND CONCLUSION

We have built a domain-dependent solution for immunization cold chain management on top of the ODK-X domain-independent framework. This allows us to leverage reusable code and a domain-dependent solution can make it easier for organizations to use by 1) shrinking the number of options that have to be configured for the system to be used 2) minimizing the vocabulary being used to the domain space so the system features are more intuitive to users, and 3) having a set of predefined roles (e.g., immunization technicians, supervisors, ministry of health officials) that enable customized interfaces that display the appropriate information to the role. Cloud hosting is becoming a common choice for *Global Goods* software creating an important issue around the location of the data and who has credentials to manage the server and control access to the data.

We now compare the approach that we have taken to several other *Global Goods* systems: DHIS2, OpenMRS, and OpenLMIS. There are a set of common considerations across these *Global Goods* and our system, namely the need for a mechanism to customize individual instances for country or local deployment, and the desire to protect the integrity of the code base so that individual instances do not result in forking the platform.

One of the reasons that DHIS2 has been so successful is that it can be customized at the country level for reporting data up through an arbitrary hierarchy. This is provided through the core data structures of an “Org unit hierarchy” with user defined reporting attributes. This model requires a moderate level of training for implementers but gives a fairly general solution. The deployment model is a country level implementation team to manage the reporting system. This transfers the data acquisition problem we faced to the country team, but potentially simplifies the task by moving it closer to the MoH. Our approach differs from DHIS2 in that we are relying on directly creating per country instances of a more specialized application.

OpenLMIS provides multiple levels of customization. Forms and reports can be updated through configuration files or through a wizard. OpenLMIS is built using a service based architecture, so the choice of components can be customized. The deployment model for OpenLMIS include build scripts for generating the different modules used by countries. Logistics requires custom modules to handle implementation of different replenishment strategies so there is a substantial amount of domain knowledge that must be implemented in code. The major difference between OpenLMIS and our system is that OpenLMIS targets compiling a separate application per country, in contrast to out goal of having the same application for each country.

Finally, OpenMRS is a platform for building medical record systems, which can either be modifications of a reference implementation built from scratch. A central aspect of OpenMRS is its “concept dictionary” which allows definition of all terms and attributes in the application. OpenMRS also has a very general module structure to allow custom extensions to be created. In contrast to our work, is a platform for building local medical record applications, in contrast to a system design for multiple, uniform deployments.

This brief review of other global goods system places the Cold Chain Application at one end of a design space which generates a single application to be used across a large number of countries. DHIS2 and OpenLMIS have a middle position in the spectrum, of focusing on country level customization, while OpenMRS targets a more local level of customization. One reason that we can take a global approach is that our tool has a narrower scope than these other *Global Goods*.

REFERENCES

[1] 2020. FrontlineSMS. (2020). https://www.frontlinesms.com/.

[2] 2020. GADM maps and data. (2020). http://gadm.org/.

[3] 2020. Mobile Operating System Market Share Worldwide, February 2020. https://gs.statcounter.com/os-market-share/mobile/worldwide. (March 2020).

[4] 2020. Principles for Digital Development. (2020). http://digitalprinciples.org/.

[5] 2020. The Humanitarian Data Exchange. (2020). https://data.humdata.org/.

[6] 2020. Ushahidi. (2020). https://https://www.ushahidi.com/.

[7] Jørn Braa, Ole Hanseth, Arthur Heywood, Woinshet Mohammed, and Vincent Shaw. 2007. Developing Health Information Systems in Developing Countries: The Flexible Standards Strategy. *MIS Q.* 31, 2 (June 2007), 381–402.

[8] Jørn Braa and Calle Hedberg. 2002. The Struggle for District-Based Health Information Systems in South Africa. *The Information Society* 18, 2 (2002), 113–127. DOI:http://dx.doi.org/10.1080/01972240290075048

[9] Jørn Braa and Sundeep Sahay. 2017. The DHIS2 Open Source Software Platform: Evolution Over Time and Space. In *Global Health Informatics. Principles of eHealth and mHealth to Improve Quality of Care*, Leo Anthony, G. Celi, Hamish S. F. Fraser, Vipan Nikore, Juan Sebastian Osorio, and Kenneth Paik (Eds.). The MIT Press.

[10] Waylon Brunette, Samuel Sudar, Mitchell Sundt, Clarice Larson, Jeffrey Beorse, and Richard Anderson. 2017. Open Data Kit 2.0: A Services-Based Application Framework for Disconnected Data Management. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys 2017)*. ACM, New York, NY, USA, 440–452. DOI: http://dx.doi.org/10.1145/3081333.3081365

[11] Waylon Brunette, Mitchell Sundt, Nicola Dell, Rohit Chaudhri, Nathan Breit, and Gaetano Borriello. 2013. Open Data Kit 2.0: Expanding and Refining Information Services for Developing Regions. In *Proceedings of the 14th Workshop on Mobile Computing Systems and Applications (HotMobile '13)*. Association for Computing Machinery, New York, NY, USA, Article 10, 6 pages. DOI: http://dx.doi.org/10.1145/2444776.2444790

[12] Andrew Cross, Nakull Gupta, Brandon Liu, Vineet Nair, Abhishek Kumar, Reena Kuttan, Priyanka Ivatury, Amy Chen, Kshama Lakshman, Rashmi Rodrigues, and et al. 2019. 99DOTS: A Low-Cost Approach to Monitoring and Improving Medication Adherence. In *Proceedings of the Tenth International Conference on Information and Communication Technologies and Development (ICTD '19)*. Association for Computing Machinery, New York, NY, USA, Article Article 15, 12 pages. DOI: http://dx.doi.org/10.1145/3287098.3287102

[13] Dimagi. 2020. The CommCare Evidence Base & Overview. (2020). https://bit.ly/39vlo8S

[14] Carl Hartung, Yaw Anokwa, Waylon Brunette, Adam Lerer, Clint Tseng, and Gaetano Borriello. 2010. Open Data Kit: Tools to Build Information Services for Developing Regions. In *Proceedings of the 4th ACM/IEEE International Conference on Information and Communication Technologies and Development (ICTD '10)*. Association for Computing Machinery, New York, NY, USA, Article Article 18, 12 pages. DOI: http://dx.doi.org/10.1145/2369220.2369236

[15] Richard Heeks. 2002. Information Systems and Developing Countries: Failure, Success, and Local Improvisations. *The Information Society* 18, 2 (2002), 101–112.

[16] Richard Heeks. 2006. Health information systems: Failure, success and improvisation. *International journal of medical informatics* 75 (03 2006), 125–37. DOI: http://dx.doi.org/10.1016/j.ijmedinf.2005.07.024

[17] F Huang, S Blaschke, and H Lucas. 2017. Beyond pilotitis: taking digital health interventions to the national level in China and Uganda. *Globalization And Health* 13, 1 (2017).

[18] Klaus Krickeberg. 2007. Principles of Health Information Systems in Developing Countries. *Health Information Management Journal* 36, 3 (2007), 8–20. DOI:http://dx.doi.org/10.1177/183335830703600303

[19] Henry Lucas. 2008. Information and communications technology for future health systems in developing countries. *Social Science & Medicine* 66, 10 (2008), 2122–2132.

[20] Joseph Maina, Paul O. Ouma, Peter M. Macharia, Victor A. Alegana, Benard Mitto, Ibrahima Socé Fall, Abdisalan M. Noor, Robert W. Snow, and Emelda A. Okiro. 2019. A spatial database of health facilities managed by the public health sector in sub Saharan Africa. *Scientific Data* 6, 1, Article 134 (2019).

[21] Burke W Mamlin, Paul G Biondich, Ben A Wolfe, Hamish Fraser, Darius Jazayeri, Christian Allen, Justin Miranda, and William M Tierney. 2006. Cooking up an open source EMR for developing countries: OpenMRS - a recipe for successful collaboration. *AMIA ... Annual Symposium proceedings / AMIA Symposium. AMIA Symposium* (2006), 529.

[22] Marc Mitchell and Lena Kan. 2019. Digital Technology and the Future of Health Systems. *Health Systems & Reform* 5, 2 (2019), 113–120. DOI: http://dx.doi.org/10.1080/23288604.2019.1583040

[23] Henry Mwanyika, David Lubinski, Richard J. Anderson, Kelley Chester, Mohamed Makame, Matt Steele, and Don de Savigny. 2011. Rational Systems Design for Health Information Systems in Low-Income Countries: An Enterprise Architecture Approach. *Journal of Enterprise Architecture* 7, 4 (2011).

[24] PATH. 2019. Digital Square Global Goods Guidebook. (2019). https://bit.ly/2SNhrGF

[25] Fahad Pervaiz, Aditya Vashistha, and Richard Anderson. 2019. Examining the Challenges in Development Data Pipeline. In *Proceedings of the 2nd ACM SIGCAS Conference on Computing and Sustainable Societies (COMPASS '19)*. Association for Computing Machinery, New York, NY, USA, 13–21. DOI: http://dx.doi.org/10.1145/3314344.3332496

[26] Johan Saebo and Ola Titlestad. 2004. Evaluation of a bottom-up action research approach in a centralised setting: HISP in Cuba. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences,*, Vol. 37. 11 pp. DOI: http://dx.doi.org/10.1109/HICSS.2004.1265384

[27] Sundeep Sahay and John Lewis. 2010. Strengthening Metis Around Routine Health Information Systems in Developing Countries. *Information Technologies & International Development* 6, 3 (2010).

[28] Sundeep Sahay, Eric Monteiro, and Margunn Aanestad. 2009. Toward a political perspective of integration in information systems research: The case of health information systems in India. *Information Technology for Development* 15, 2 (2009), 83–94. DOI: http://dx.doi.org/10.1002/itdj.20119

[29] USAID and WHO. 2018. Master facility list (MFL) resource package: Guidance for countries wanting to strengthen their MFL. (2018). https://www.who.int/healthinfo/MFL_Resource_Package_Jan2018.pdf

[30] VillageReach. 2012. The Framework for OpenLMIS. (February 2012). http://www.villagereach.org/wp-content/uploads/2012/03/02292012.Framework-for-OpenLMIS-Whitepaper.pdf

[31] WHO. 2013. Creating a Master Health Facility List. (2013). https://bit.ly/2IvRW6K

[32] M. Zaffran, J. Vandelaer, D. Kristensen, B. Melgaard, P. Yadav, K. O. Antwi-Agyei, and H. Lasher. 2013. The imperative for stronger vaccine supply and logistics systems. *Vaccine* 31 (April 2013), B73–B80.