# Open Data Kit: Technologies for Mobile Data Collection and Deployment Experiences in Developing Regions

Carl Hartung

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2012

Reading Committee:

Gaetano Borriello, Chair

Richard Anderson

Alan Borning

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

**Abstract**

Open Data Kit: Technologies for Mobile Data Collection and Deployment Experiences in Developing Regions

Carl Hartung

Chair of the Supervisory Committee:
Professor Gaetano Borriello
Computer Science and Engineering

Gathering information accurately and quickly is essential for enabling organizations working in low-resource settings to have timely and sustainable impact. Due to insufficient infrastructure, many organizations currently use paper to collect data in the field, only to have data entry clerks digitize the data later. This often introduces latency and potential sources of error. However, the growing development of cellular infrastructure combined with the rapid decline in the cost of smart phones presents an opportunity to shift the primary collection medium from paper to mobile devices. This dissertation presents our contribution to data collection in developing regions, Open Data Kit (ODK), an extensible, open-source suite of tools designed to facilitate tasks at every level of data collection campaigns. ODK currently provides three tools to this end: Collect, Aggregate, and Build. Collect is a mobile client providing simple interfaces for collecting data. Aggregate is an easy to deploy data storage system hosted in the "cloud" or on local servers. Build is a web-based drag-and-drop form designer created to simplify the process of creating complex digital forms. By providing the ability to both capture and present richer data (e.g. images, video, and location), ODK tools have provided organizations new ways to collect and analyze information. We present the system architecture and through example real-world de-

ployments, highlight specific design decisions that have enabled new directions in data collection and workforce management. Finally, we discuss lessons learned in building the system and present promising future directions in the space.

# TABLE OF CONTENTS

Page

List of Figures . . . . . iii

List of Tables . . . . . v

Glossary . . . . . vi

Chapter 1: Introduction . . . . . 1
1.1 Data Collection Campaigns . . . . . 1
1.2 Development Scenarios . . . . . 5
1.3 Paper . . . . . 8
1.4 From Paper to Technological Solutions . . . . . 9
1.5 Facilitating the Creation and Operation of Mobile Information Systems . . . . . 20
1.6 Contributions . . . . . 23
1.7 Role . . . . . 24
1.8 Summary . . . . . 25

Chapter 2: Open Data Kit . . . . . 27
2.1 XForms: A Common Format . . . . . 27
2.2 ODK: Components . . . . . 37
2.3 Operation of the Toolkit Components . . . . . 45
2.4 Summary . . . . . 50

Chapter 3: Extensibility and Modularity . . . . . 51
3.1 Form Design Modules . . . . . 51
3.2 Server Modules . . . . . 52
3.3 Mobile Client Modules . . . . . 54
3.4 Community Modification and Customization . . . . . 56

3.5 Community Support . . . . . 63
3.6 Summary . . . . . 63

Chapter 4: Initial Deployment Examining Feasibility and Training *Grameen Application Laboratory, Uganda* . . . . . 65
4.1 Background . . . . . 66
4.2 Implementation . . . . . 70
4.3 Results . . . . . 76
4.4 Summary . . . . . 81

Chapter 5: Deployment 2: Examining Cost, Extensibility, and Integration with Existing Systems *Academic Model for the Prevention and Treatment of HIV, Kenya* . . . . . 83
5.1 Background . . . . . 84
5.2 Implementation . . . . . 88
5.3 Results . . . . . 92
5.4 Summary . . . . . 94

Chapter 6: Deployment 3: Evaluating Supervision, Data Quality, User Interaction, and Multimedia *Action Research and Training for Health, India* . . . . . 96
6.1 Background . . . . . 97
6.2 Implementation . . . . . 107
6.3 Results . . . . . 120
6.4 Summary . . . . . 127

Chapter 7: Conclusion . . . . . 130
7.1 Retrospective . . . . . 132
7.2 Future Work . . . . . 137
7.3 Summary . . . . . 141
7.4 Acknowledgements . . . . . 142

Bibliography . . . . . 143

Appendix A: User Event Logging for ARTH Deployment . . . . . 155

# LIST OF FIGURES


| Figure Number | | Page |
|---|---|---|
| 1.1 | Even the most organized paper systems can be daunting to search. . . | 10 |
| 1.2 | Real world example of patient information stored in paper ledgers. . . | 10 |
| 1.3 | Screenshot of Cybertracker running on a PDA . . . . . | 12 |
| 1.4 | Screenshot of Javarosa running on a feature phone. . . . . | 16 |
| 1.5 | Filling out a form using CAM. (Image from [92]) . . . . . | 17 |
| 1.6 | Example of a bubble form readable by mScan. (Image from [36]) . . . . | 19 |
| 2.1 | Example of a basic XForm. The beginning and the end of the Instance section is highlighted in red, the Bindings section is highlighted in blue, and the Body section is highlighted in green. . . . . | 29 |
| 2.2 | Sample XForm and a corresponding *instance* representing a completed form. . . . . | 30 |
| 2.3 | Example Bindings showing 1) a constraint that a multi-select question not have both option "c" and option "d" selected at the same time. 2) That the current question must have a value less than 10. 3) That the current answer must be a date after the current date. . . . . | 32 |
| 2.4 | itext example . . . . . | 34 |
| 2.5 | media example . . . . . | 35 |
| 2.6 | Examples of using the appearance tag . . . . . | 36 |
| 2.7 | In ODK Build, prompts appear on the left of the screen while properties appear on the right. Users rearrange prompts using a drag and drop interaction in the web browser. . . . . | 38 |
| 2.8 | ODK Aggregate submissions in its default table format. Each row is a completed form, and each column represents the data variable from each form. . . . . | 41 |
| 2.9 | Examples of common widgets in Collect . . . . . | 44 |

2.10 The current components of ODK. Build, a form designer that generates XForms containing the logic for user interaction and for data store creation. Collect, a client that renders the form on a mobile and sends entered data to Aggregate, the server for data storage. . . . . 47

3.1 Example Intents to take a picture and to select a picture from a gallery. 55

3.2 Grameen's integration of ODK with SalesForce . . . . . 58

4.1 Phones used by SPOs before ODK . . . . . 68

4.2 The initial version of ODK Aggregate that provided a tabular view of the information, access to images, and ability to create maps in Google Maps. . . . . 71

4.3 G1 phone used by SPOs testing ODK Collect. . . . . 73

4.4 Map showing locations of SPOs working with Grameen in Uganda, built from survey responses about what crops farmers are growing. . . 74

4.5 Training SPOs using a projection of the Android Emulator . . . . . 75

4.6 Timeline of the Grameen deployment . . . . . 76

5.1 A map of AMPATH's catchment area where it provides treatments with 26 parent and 26 satellite clinics. . . . . 85

5.2 Palm TX, eTrex GPS, and connecting cable used by AMPATH prior to implementing a system with ODK. . . . . 86

5.3 A HIV counselor from AMPATH in Kenya scans a patient's barcode with ODK Collect before sending data to the OpenMRS medical record system. . . . . 90

5.4 Timeline of the AMPATH deployment . . . . . 92

6.1 A page from ARTH's PNC data collection and diagnosis form. . . . . 100

6.2 Screenshots of new widgets created for ARTH . . . . . 110

6.3 PNC form in Collect using images for Hindi script. . . . . 114

6.4 Our setup for creating the local videos with messages about taking iron, proper breastfeeding, and keeping the baby warm. . . . . 116

6.5 Timelines from our deployment with ARTH representing our planned and actual timeline. . . . . 119

6.6 Curious community members join in watching the educational videos. 128

# LIST OF TABLES

Table Number Page

5.1 Responses to an 11-question survey posed to 57 community health workers of the HCT program in western Kenya . . . . . 95

6.1 Comparing mothers who did and did not receive counseling information about the given topics with whether they reported having received it or not. n = 161 . . . . . 106

# GLOSSARY

ANDROID: An open source operating system for smart phones managed by Google.

BASIC PHONE: Phones that support only standard phone features such as voice calls and text messaging.

FEATURE PHONE: Second tier of phones, slightly more advanced than Basic Phones. Can run applications, usually written in J2ME, but lack the processing capability, advanced features, and interfaces found in Smart Phones.

GPS: Global Positioning System

SMART PHONE: Advanced phones usually containing GPS, internet connectivity, touch screens, and advanced processing capabilities.

SMS: Short Messaging Service. The protocol for sending text messages between phones.

XFORM: An XML format specifying data model, navigation, and presentation of forms.

XML: Extensible Markup Language. A markup language that defines a set of rules for encoding documents.

# Chapter 1
# INTRODUCTION

With the rise of the international development movement, the ability for organizations to gather, organize, analyze, and present data has become critically important. Spurred by the Millennium Development Goals [74], officially established by the United Nations following the Millennium Summit in 2000, organizations ranging from non-profits to governments have undertaken a huge variety of projects to improve healthcare, reduce poverty, and increase the standard of living for the estimated 1.4 billion people in the world living on less than $1.25 a day [28]. Many of these organizations primarily rely on surveys as their principal means of collecting data from these low-income populations [118].

## *1.1 Data Collection Campaigns*

For organizations working on projects in developing regions, collecting and analyzing data in a timely, cost effective, and accurate fashion is central to all aspects of their interventions [94, 123]. First, identifying where and what problems to tackle requires good data about the issues facing the local communities. Once a project has begun, determining the effectiveness of the various aspects of the programs depends upon the organization's ability to document changes influenced by the interventions. Finally, combining the data, organizing and understanding it, and using it to generate reports provides the public and donors a window into the organization's overall impact and influences future project and funding opportunities.

For any given data collection campaign, the process can be separated into three distinct phases: 1) determining what information to collect and how to collect it,

2) gathering data from the field field, and 3) cleaning, aggregating, and analyzing the data. Initially, projects must follow the process in order, but after the system is bootstrapped all three phases can occur simultaneously, potentially greatly influencing each other.

### *1.1.1 Determining the Information to Collect*

The first component of any data collection campaign is determining what information to gather and how to go about capturing it. From phone calls to door-to-door campaigns, a huge variety of work exists highlighting the pros and cons of each method [60]. Since our work focuses on organizations working in regions with limited resources, we focus on campaigns requiring organizations to send workers into the field to gather data.

Although the details of how questions are phrased and even how they are ordered are critically important in the design of any campaign, deciding the how and the what of an intervention are outside the scope of this dissertation. We instead focus on giving organizations the ability to create these campaigns, and leave the exact details to the implementors.

### *1.1.2 Data Gathering*

The next component of any data collection campaign is the actual collection of the information itself. Organizations send workers into the field armed with forms specifying what information to gather about the target subject. The exact number of workers can vary greatly and ranged from five to almost 4,000 in deployments with which we were directly involved. Similarly, the length of time workers spent in the field varied from one day to several months depending on the goals and organization of the task. Organizations will either task additional workers to find those working in the field and collect their data, or simply wait for the data collectors to return with their information.

While collecting data may be the primary objective, in many cases the field workers are also tasked with using the information gathered in situ to make locally relevant decisions and take immediate action. Often the instructions for this data-driven decision support is explicitly built into the forms and includes instructions such as administering a certain amount of a given medication if a specific combination of symptoms are observed. The use of forms for data-driven decision making is especially important in developing regions where access to highly trained experts may be unavailable [124] and organizations need to rely on more *lightly trained* workers [76].

Lastly, field workers are often used to deliver information in addition to gathering data. In this way, they not only become interviewers, observers, and recorders, but educators as well. For example, an agricultural worker collecting information about pests and diseases affecting local crops may also be tasked with distributing information about how to combat those pests along with regionally relevant farming techniques to improve crop yield. It is important in these contexts that the worker can effectively convey the information in a trusted manner.

### *1.1.3 Data Aggregation and Analysis*

After the data has been collected it is often passed on to data entry or data quality clerks. These workers review the data, check for errors, typically enter it into an electronic system, and file away any artifacts. During their review process they may contact the data collectors to clarify entries that cannot be interpreted, appear wrong, or that they do not understand.

After the data clerks enter the data into an electronic system, data managers or administrators can use the data to make high-level decisions related to the interventions and generate reports for donors and/or the public.

As the process unfolds, each step may influence the how, where, and what of the others. For example, detection of many errors at the aggregation step may indicate that the workers gathering the data need some clarification about the process. Sim-

ilarly, administrators analyzing the aggregated data may notice trends or unusual results in specific areas and re-task the workers towards those regions.

### *1.1.4 Open Data Kit*

Currently, paper is the most common medium used to implement data collection campaigns in developing regions. While paper has the benefits of being inexpensive and universally available, often it comes with hidden costs such as the time to enter the data into an electronic system. Furthermore, researchers have shown that paper can be extremely error prone, and in many cases requires prolonged periods of time (weeks to months) to aggregate and analyze [26]. To combat these issues attempts have been made to digitize the entire process using various technologies such as PDAs, basic phones with voice and SMS, and feature phones running applications. These systems have enjoyed some degree of success, but often their tightly coupled, closed, and monolithic design has prevented long term sustainability.

To address these disparities, we have leveraged the next steps in technological advancement and implemented a data collection system, Open Data Kit (ODK) [7] using smart phones and cloud technologies targeted at solving the issues commonly faced by organizations collecting data in developing regions. ODK provides a flexible, modular design allowing for customization, integration with existing systems, and extensibility targeted at compatibility with future systems. We show through example real-world deployments how the features enabled by ODK provide project implementors previously unavailable information about their interventions. Furthermore, we show ODK can reduce costs, improve timeliness of data, and increase data quality.

The remainder of this chapter considers various development scenarios and the problems often faced by organizations attempting data collection campaigns in developing regions. We discuss the benefits and limitations of previous technological solutions, providing the framework from which we developed our system. Finally, we outline the contributions of ODK to the space of mobile data collection.

### *1.2 Development Scenarios*

To demonstrate the types of interventions targeted by this dissertation, we present an example. Imagine a community health organization working in rural Africa. This organization has several goals: to gather statistics about the prevalence of HIV, TB, and malaria, to treat as many of the affected people as possible, to educate the population about how different diseases are transmitted, and to build a database so that patients and their health care providers have complete medical records. To accomplish these goals, the organization has hired community health workers to travel through villages, meet with residents, administer voluntary testing and counseling, and record the results.

In this scenario, both the health worker and the organization have a different set of needs:

**The Health Worker Needs:**

- a way to record and/or retrieve patient information;
- training materials to educate the patients;
- knowledge of when to administer specific tests, and testing materials; and
- a way to deliver the collected information to the central clinic.

**The Organization Needs:**

- to train health workers how to register new patients and record follow up visits and test results;
- to collect the forms and digitize them in a timely manner;
- a place to store all of their data so it is secure and easily retrievable; and
- to potentially scale from ten health workers to hundreds, and from tens of patients to millions.

This community health example is what we consider a data collection campaign. Data collection campaigns implemented by organizations typically involve many field-workers traveling around gathering and disseminating information as part of an intervention targeted at improving some metric such as healthcare or crop yield. The data collected can both help the organization make decisions on a project level, and also provide services such as decision support to the worker at the time of the data collection. To further illustrate examples of data collection campaigns, below is a list of real tasks from organizations that the authors interacted with over the course of the work encompassing this dissertation.

**Examples of Real World Data Collection Campaigns:**

- Government workers completing socio-economic surveys about households in a district.

- Agricultural extension workers creating an application with video and audio clips explaining farming techniques.

- Teachers implementing games with interactive questions and answer tutorials and automatic score recording.

- Crisis workers capturing images and locations of damaged areas after an earthquake.

- Funders receiving geo-tagged reports of interventions they have supported.

- Clinicians building decision support applications that use patient data to help determine when to administer tests.

- Microfinance institutions tracking transactions from lenders and borrowers.

- Indigenous tribes cataloging their trees to both create public awareness about illegal logging as well as enable participation in global carbon markets.

- Community health workers collecting medical data while diagnosing and treating pregnant women during household visits.

### *1.2.1 Data Collection Issues*

Though the exact types of organizations, workers, and tasks vary widely across domains, cultures, and locations, the processes, challenges, and solutions can be generalized under the category of data collection campaigns and represented by the three phases discussed in Section 1.1. Through working with organizations trying to solve the problems listed above, the following challenges emerged as common themes when attempting to implement data collection and dissemination campaigns in developing regions:

- **Cost:** Keeping costs low is crucial to many projects. Frequently projects are undertaken by non-profit Non-Governmental Organizations (NGOs), funded strictly by donations from outside sources, and many utilize mostly unpaid volunteers to help complete their work.

- **Time:** Minimizing the time it takes for information to flow from the point of capture to administrative decision makers can be crucial for projects to have relevant impact. Decreasing this delay makes projects more agile and can reveal errors in the process more quickly.

- **Supervision:** Workers are often in remote areas for weeks at a time, making supervision very difficult. Without proper supervision and support, workers may make repeated errors or simply fail to complete their tasks.

- **Training:** Organizations, especially those relying heavily on volunteers, often have high turnover rates and frequently need to train new workers. Since training incurs cost and requires time, making training easier and more efficient decreases both cost and time.

- **Data Quality:** For an organization, knowing that they have collected real, accurate data is invaluable to the success of their projects. High-level project decisions are often data driven, and inaccurate or falsified data can cause the administrators to make decisions that result in less positive impact.

The issues of cost, time, supervision, training, and data quality were mentioned by every one of our partner organizations. As such, we used these as our primary criteria in examining existing systems and also when evaluating our own contributions.

### *1.3 Paper*

As we began working with organizations implementing data collection campaigns, we found that many used paper as their primary means of information gathering, decision support, and outreach despite well documented inefficiencies [112]. Organizations' use of paper often stemmed from legacy projects, but some cited cost, difficulties in use or training, and lack of capabilities in current technology systems as reasons for not attempting the jump to a more advanced solution.

The common use of paper in data collection campaigns was not surprising as paper has many inherent advantages. Paper is inexpensive, universally available, and requires very little training for a literate worker. However, the properties that make paper simple to use also have some large disadvantages, listed below.

**Timeliness:** The first problem organizations reported with paper was the lag from the collection of the data to having it an actionable format. With field workers often set in remote regions, organizations either had to spend significant amounts on transportation to gather the data, or incur a delay until the worker returned with the

collected information. Also, once the paper had been gathered at a central location, manually digitizing the data required even more time.

**Error Prone:** The second issue reported was related to data quality. For large projects, taking the data from the paper and entering it into an electronic system can be very time consuming, man-power intensive, and create numerous opportunities for the introduction of errors. Data entry clerks often had to interpret the data collectors' handwriting and/or could introduce errors through typographic mistakes. To combat this, organizations often used a double-entry system where two people would enter the data into the system, the entries would be compared, and discrepancies would be manually sorted out by a manager [105, 35, 78]. While double-entry helps to catch some of the data errors, it introduces even more cost and time into the process.

**Physical Artifacts:** Other problems cited by organizations were that records were sometimes lost, accidentally discarded, or destroyed before entry. Furthermore, paper filing systems were hard to search (see Figure 1.1) and impossible to transport or back up (Figure 1.2).

Our observations led us to conclude that paper-based data collection systems limit the scale and complexity of the services that can be provided, and thus the impact of the interventions.

### *1.4 From Paper to Technological Solutions*

Though personal computers have been popular since the mid 1990s, and laptops have been common since the mid 2000s, few organizations have successfully integrated these technologies into remote field work. Systems targeted at low income regions have an additional set of hardship constraints that the designers must consider [19, 83]. These hardships include, but are not limited to: infrastructure problems such as intermittent power and internet connectivity [120], cultural sensitivities [9, 109], low education and literacy rates [71], and extreme environmental conditions such as heat, rain, and dust.


Figure 1.1: Even the most organized paper systems can be daunting to search.

Figure 1.2: Real world example of patient information stored in paper ledgers.

Of course, computing technology is no panacea, as noted by Toyama et al. [116] and Brewer et al. [19], but with the growth of mobile phone usage in these regions [70] there have come opportunities to digitize and automate many of these data collection campaigns in a cost effective manner.

Battery powered mobile devices with wireless connectivity help alleviate some of the constraints caused by limited access to reliable electricity and wired internet connections. Proliferation of mobile phones and cellular coverage creates local familiarity and expertise with the devices. Finally, improved smart phone technology has put a tremendous amount of computational power in a lightweight, easily transportable form factor.

The remainder of this section examines the evolution of technological solutions for data collection, the issues and limitations revealed by prior deployments, and reasons for the success or failure of various systems.

### *1.4.1 From Paper to PDAs*

In an attempt to assuage some of the constraints of paper-based systems, organizations first experimented with PDA-based technologies [61, 46]. Initial systems that emerged focused on specific purposes such as collecting data for Malaria [50] or providing decision support for paramedics working in remote regions [5]. Later, more generic systems emerged such as the free and open-source EpiHandy [117] and the commercially available Pendragon Forms [99].

One of the first examples of PDA-based data collection is CyberTracker [33], a system first developed in the mid-1990s as a way to enable non-literate animal trackers to record observations on PDAs (sometimes with attached GPS units) using a purely graphical and non-linear interface. Trackers, when observing a specific animal behavior, tapped a representative icon on the screen to mark that behavior as shown in Figure 1.3. For applications such as socio-economic surveys, CyberTracker replaced animal behavior icons with icons representing families, houses, and marriage status.

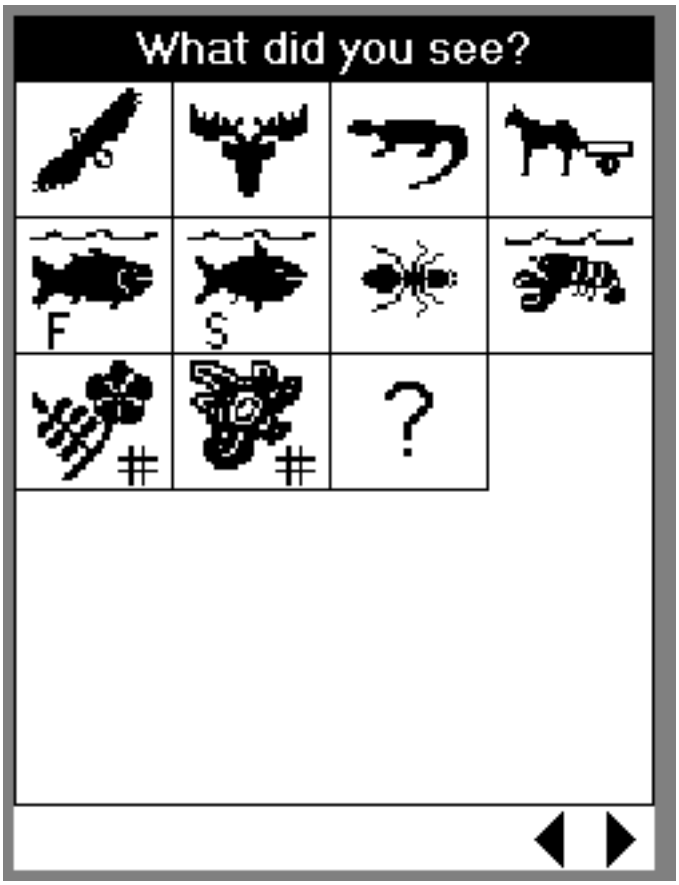



Figure 1.3: Screenshot of Cybertracker running on a PDA

CyberTracker is still in wide use today and has added functionality including a form designer, data synchronization over the web, and image capture. While these upgrades build toward a more generic system, they do not change the fundamental use case and interactions. That is, CyberTracker is designed for gathering large quantities of geo-referenced data from illiterate field observers and synchronizing those observations to a local computer.

For broader use cases than CyberTracker targets, Pendragon Forms [99, 100] has been a popular and fully-featured commercial solution that includes a form designer, data synchronization, and forms with navigation logic. Pendragon is limited by its cost at $80 per user, and customizing or extending the application is impossible since the software is closed-source. Although designed for developed regions, Pendragon Forms has also been used all over the world [16, 112] for collecting data on households [113], tuberculosis [17, 15], and patients with respiratory tract infections [41].

Studies about the effectiveness of PDAs show mixed results. Multiple studies in both developed and developing world settings have shown that using PDAs can reduce the amount of time spent documenting and collecting information [18, 67, 107, 80]. However, studies using PDAs for protocol navigation showed the same to slightly longer times versus current practice [40]. This could be attributed to lack of familiarity with the technology, and also that current practices in developing regions rarely accurately follow the protocols whereas the PDAs did not provide as much flexibility to skip steps. Workers using the technology noted the importance of making sure the technology did not increase time spent on tasks [39].

Similarly, other studies have shown contradictory results about the appropriateness of PDAs as a platform for the developing world. Cheng et. al found that people tended to give more socially desirable answers when the interviewer used a PDA [29], while Kurth et al. showed that PDAs were the preferred medium [79]. These studies were carried out in very different cultures (Angola and Peru, respectively), so local context may have played a factor in the results.

Two major reviews of the use of mobile devices in healthcare found that though there was "evidence to both support and refute" mobile devices as an effective healthcare platform [77], "most care providers found PDAs to be functional and useful..." [82]. Additionally, PDAs have been shown to be extremely useful for large scale data collection. In Mtwara, Tanzania PDAs were successfully used to survey over 270,000 households [113] for baseline health information. Similarly, PDAs were used in nationwide efforts in both Togo and Niger to determine the coverage of bed nets throughout each country [45].

Though implementing data collection on PDAs looked promising as they were shown to be less expensive and less error prone than pen and paper [95], the systems developed for these platforms were often task-specific and siloed, that is, every component of the system depended on every other componenet. As mobile phones started eclipsing PDAs in popularity, manufacturers stopped developing and supplying PDAs. Unable to replace devices as they broke down, the resulting attrition rendered these inflexible systems useless and forced organizations to seek other alternatives.

### *1.4.2 From PDAs to Mobile Phones*

With the nearly exponential growth of mobile phones in developing regions [72], phones have replaced PDAs as a preferred platform for data collection due to their more flexible connectivity options [115]. However, many of the lessons learned from the initial PDA systems are still valuable for newer mobile phone based systems.

Initially, organizations tried to take advantage of the proliferation of mobile phones by creating call-in systems that used voice prompts, recorded audio, and touchtone input [111, 32, 98]. Others have relied on SMS messages to transmit information [53, 104, 13]. Feature phones add the ability to run small applications and some add a camera for taking photographs. Several organizations have created data collection applications specifically designed for feature phones [85, 52]. The newest generation of phones, termed *"smart phones"*, go a step beyond feature phones and

add real processing capabilities, advanced interfaces like touch screens and/or full keyboards, and an array of sensors such as GPS, compass, and three dimensional accelerometers. Our work targets smart phones because they are the only phones with the ability to process large, complex protocols, display high resolution photographs, and play videos. Due to the higher cost of smart phones, we built our system targeting structured organization-level data collection rather than individual or "crowd" use cases.

For data collection software, there are free and/or open-source competitors to Pendragon Forms implemented on feature phones running Java Platform, Micro Edition (J2ME) such as FrontlineForms [53], EpiSurveyor [47], CommCare [30, 85], and JavaRosa [73] (shown in Figure 1.4), which have become popular as the prices of Java-enabled phones have fallen. Unfortunately, these phones live in a fragmented ecosystem that negatively impacts software development and usability.

J2ME applications must often be signed by the vendor, carrier, or manufacturer before interactions with storage, networking, or hardware accessories are usable. Without the appropriate digital certificates and signatures, users are prompted with confusing dialogs before every such action. The signing process can require months of waiting and thousands of dollars. Even after signing authority is obtained, capturing images, audio, video, and location remains difficult because each device implements the interface to its underlying hardware differently. J2ME programmers are forced to test every software release on each physical device they wish to support – a requirement that nullifies many of the benefits of a wide phone base.

Like Pendragon Forms, many of these J2ME-based systems do not support the free flow of information that is captured. For example, when using FrontlineForms on a phone a user can only transfer data to/from a PC running FrontlineSMS. Furthermore, FrontlineForms uses a proprietary format that makes it difficult for organizations to transfer data using another mobile client or service.

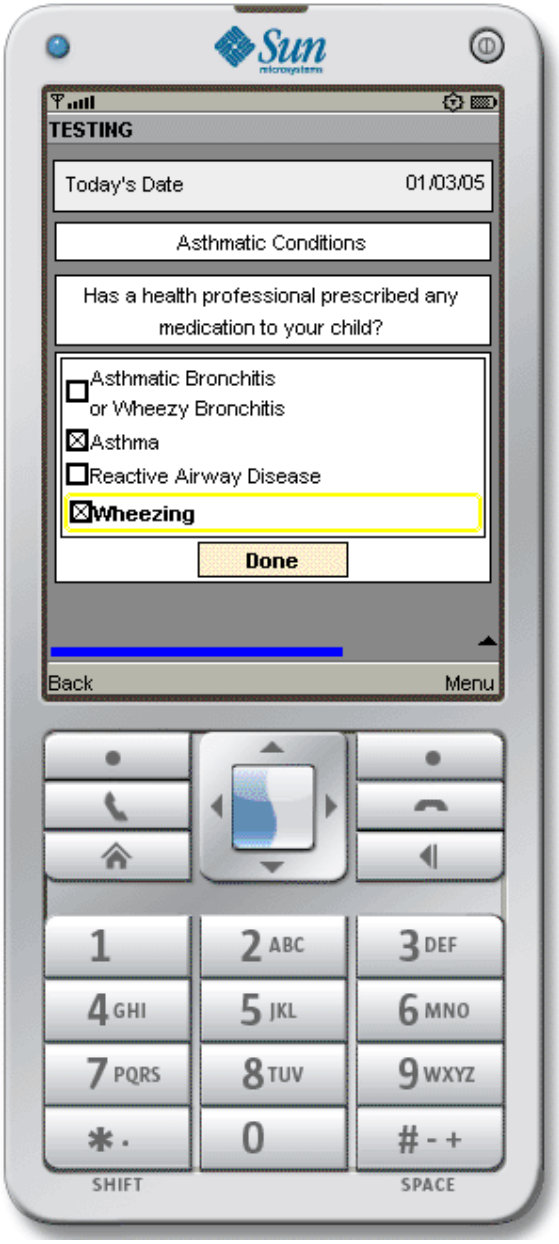



Figure 1.4: Screenshot of Javarosa running on a feature phone.

Figure 1.5: Filling out a form using CAM. (Image from [92])

In the academic literature, the best example of data capture and survey software is Froehlich et al.'s MyExperience [51]. MyExperience gathers objective data such as user context (as sensed by the device, e.g., current location) along with sensor readings. It uses context-triggers to capture in situ subjective user feedback. MyExperience's use case is to collect quantitative and qualitative data in the field to support studies of mobile technology usage and evaluation and, more recently, psychological studies of human attitudes and behavior. However, there is no focus on building information services that deliver information or managing the specific challenges of developing regions.

Several systems have recently been introduced to augment paper processes with technology such as CAM [95], Shreddr [27], and mScan [36]. CAM, seen in Figure 1.5, was one of the first examples of an information service toolkit designed for developing regions. Primarily used for data collection, users fill out paper forms first and then use the phone's camera to scan barcodes to trigger data entry and submission. Key elements of the system include a close linkage to paper forms, an image and audio driven user-interface, support for both asynchronous or synchronous connectivity, and a scripting language that describes form logic. While CAM's procedural scripting makes it easy to design iterative constructs in the forms, declarative languages offer more in the way of validation and optimization. With the more lightly trained workforces often found in developing regions, the ability to do input verification in real-time can prove invaluable [25]. CAM was initially created for micro-finance applications in India, but has since been deployed in other areas such as helping improve efficiency and monitoring operations at coffee plantations in Guatemala [108, 93].

Taking a similar approach to CAM, Shreddr [27] is another system for digitizing data from paper forms. With Shreddr, a user scans a form and manually identifies the locations of each field. Data managers can then scan completed forms, Shreddr aligns the forms, segments the fields appropriately, and exports the tasks of recognizing and recording the answers in each field to people via a crowd-sourcing platform.

mScan [36] automates the process of digitizing data on paper forms by leveraging the increased processing and imaging capabilities of smart phones. mScan uses advanced image processing techniques to identify answers recorded on paper forms that use a multiple choice or bubble format, seen in Figure 1.6. An advantage of mScan over CAM and Shreddr is that it does not require re-designing of forms or manual identification of fields to enable digitization.

The hybrid paper/phone systems just discussed solve the specific problem of digitizing data that has already been collected on paper. In our work, we aim to completely replace the paper process throughout the entire pipeline of a data collection

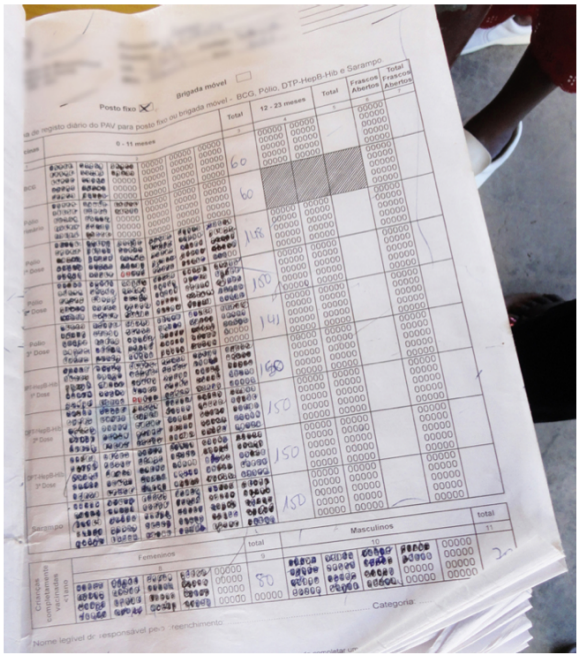


Figure 1.6: Example of a bubble form readable by mScan. (Image from [36])

campaign by digitizing data at the point of capture. Given the variety of organizations, cultural differences, and technical capacity, we believe both types of systems will be in demand for the foreseeable future.

### *1.4.3 Summary*

In examining current and past systems, we found that many had similar limiting factors. They were built as monolithic, tightly coupled solutions using proprietary data formats and interfaces. If ever one component of the system became obsolete, the whole system failed to function. Adding new features or customizing the software was often impossible. Additionally, many were built for specific hardware platforms that companies stopped manufacturing as demand decreased. Finally, early devices tended to have minimal processing power and memory, and lacked programatic access to device sensors like GPS and embedded cameras.

## 1.5 Facilitating the Creation and Operation of Mobile Information Systems

For computing technology to truly address the information gaps in developing regions, information services must be composed by non-programmers, deployed by resource-constrained organizations, used by minimally-trained users, and remain robust despite intermittent power and connectivity. To address these challenges, we developed Open Data Kit (ODK) [62], a modular, extensible, and open-source suite of tools designed to empower users to build information services. ODK currently consists of three tools: Build, Aggregate, and Collect.

ODK Build is a drag-and-drop form designer that defines the presentation, navigation logic, and data configuration used by the tools. ODK Aggregate provides a "click-to-deploy" server that can run either locally or in the cloud. Aggregate automatically creates databases based on the forms used to collect data, and presents simple APIs for data upload, storage, and transfer to other systems. Finally, ODK Collect is a mobile platform that renders the forms, automatically navigates form logic, and supports the manipulation of data types that include text, location, images, audio, video, and barcodes.

### *1.5.1 ODK Build*

ODK build provides users a web-based drag-and-drop graphical interface that allows even novice users the ability to create electronic forms. The goal of Build is to lower the barrier to entry to the system and simplify the process of creating forms so that even non-programmers can design their own data collection campaigns.

### *1.5.2 ODK Aggregate*

To simplify the process of data storage and management, we designed a server that can run either locally on a PC or on one of several cloud computing infrastructures

such as Google's App Engine [55] or Amazon's Simple Storage Service (S3) [3]. Many organizations lack the computing infrastructure to store and analyze large amounts of data, as well as the technical knowledge required to create and maintain a data storage system. Additionally, the costs associated with running, maintaining and scaling traditional database backends can be prohibitive.

Aggregate builds a data store per form when a user uploads each form definition. The form can then be sent to clients for user interaction, and clients can send resulting data back to the server. Users can view the results in a table format, or export the data into one of several other formats such as CSV (comma-delimited spreadsheets) for importing into statistical analysis tools, JSON (a standard serialization protocol) for external servers, or KML (mapping markup language) to be viewed on a variety of mapping software.

By leveraging the cloud version of our server, organizations can pilot a system without the burden of purchasing, configuring, and maintaining local servers. Starting as low as a few dollars a month and increasing with use, cloud services allow virtually unlimited scalability at reasonable costs. Running servers in the cloud also alleviates the challenges of power outages, finding skilled IT managers, managing backups, hardware failures, and protecting from viruses.

We realize, however, that many organizations may be hesitant to fully trust cloud computing services due to privacy or political concerns. Thus, we also designed our server with the option to run locally on a PC for those wishing to keep their data in-country and on storage under their complete control. Users can choose either approach to store their data and migrate between them at any time as the interfaces to both types of servers are identical.

### *1.5.3 ODK Collect*

For our primary client, we designed a mobile phone application that allows users to download forms, interact using touch gestures, and send information to servers

through wired or wireless connections. Mobile phones are ideally suited to the types of applications we targeted because of their small form factor, lower price relative to PCs and laptops, ability to run in disconnected environments with intermittent access to power, and almost universal familiarity amongst our typical users.

When the project began we made the controversial decision to build our application using current generation smart phones. We chose these phones for reasons of their programmability, robust feature sets, enhanced interaction modalities, and increased processing power. Current generation mobile phones have features such as built-in cameras, GPS, and touch screens. Unlike earlier generation phones, the operating systems and programming APIs of smart phones give developers unfettered programmatic access to all system components. Furthermore, memory and processors in the current generation of phones are approaching the speed of laptops, removing many of the constraints that impeded previous systems. Recently mobile phone operating systems have been extended to work on tablet computers and netbooks, meaning that the same applications that run on the phone will run on these other devices with no changes to the underlying source code.

While others considered such phones to be too expensive and unavailable in developing regions, we predicted that the trend of increasing mobile adoption and technology would, within a few years, lower cost and increase availability. So far, prices of first generation phones have continued to drop sharply and carriers in these regions have started offering multiple models of the phones for sale. For example, Huawei began offering the Ideos phone in both Kenya and India for roughly the equivalent of $80 USD, and as of May 2011 the IDEOS had sold over 60,000 units in Kenya [69].

Since current generation smart phones would likely be too expensive for many individuals living in developing regions, we targeted our tools at organizations working in these resource-constrained environments. Even so, given the current trends, we suspect that our system will be used in the future for crowd-sourcing data, or even creating local businesses using ODK for information gathering and delivery.

We built ODK focusing on the following set of requirements:

- **Modular Components**. By focusing on creating small, composable modules, we can create a system that is easier to extend and modify.

- **Open Source**. By utilizing open source software and interfaces based on open standards we can leverage a wider community of developers.

- **Sustainability**. Rather than building for a specific hardware platform, we developed our applications on systems that are likely to persist and evolve over the long-term, provide a diversity of available form-factors, and adapt to new capabilities made available by the rapid pace of innovation in this space.

Designed to be used together or independently, ODK tools build on existing open standards and empower individuals and organizations to compose services that collect and distribute information in the developing world. ODK is supported by an open-source community that has contributed training documents and language localization support, as well as additional tools.

### *1.6 Contributions*

The contributions of ODK center on how the choice of platforms and architecture design enable and encourage the growing number and variety of deployed applications. Although one can certainly cobble together many of these services with existing software and hardware, ODK has enabled these compositions to be realized in a short time, made the systems more deployable by non-programmers, and created the ability for external programmers to easily make modifications and customizations. We argue these distinctions matter and as evidence we note that parts of CyberTracker, MyExperience, CommCare, and EpiSurveyor have been ported to the ODK platform.

Specifically, the contributions presented in this dissertation are:

- Open Data Kit, an open-source platform for designing mobile information systems applications. To our knowledge ODK has been deployed on over 10,000 phones in 29 countries on 6 continents.

- An extendible, modular system design, giving organizations the ability to easily extend and customize the applications, utilize external components, and integrate with existing systems. We present examples demonstrating the advantages of our design.

- Evaluation of ODK, through real-world deployments, showing the feasibility of deployment in developing regions and demonstrating the effects of ODK on the key criteria specified by organizations: cost, time, supervision, training, and data quality.

- Discussion of new features added to Open Data Kit that go beyond current mobile and paper data collection systems such as the ability to use multimedia as first class objects and the capture and recording of user interactions.

We evaluated this work through real-world deployments with organizations working on various projects in developing regions. For each we gathered quantitative data as well as qualitative feedback about the use of each of the tools.

### *1.7 Role*

The design and implementation of ODK has involved students, developers, advisors, and even other organizations that have contributed feedback and source code used in each of the tools. My personal involvement began at the onset of the project, helping to architect the overall system design. I have had at least some part in developing portions of each of the tools or prototypes thereof, but my main role has been as the

primary developer of Collect. At the time of this writing Collect consists of around 12,000 lines of code.

In addition to developing the software, I also worked directly with Grameen, AMPATH, and ARTH on the deployments discussed later in Chapters 4, 5, and 6, respectively. Each project required from several weeks to nearly a year of preparation and communication from Seattle, where I was based, before deploying the system. I spent over three weeks in Uganda on our initial deployment with Grameen, nearly three weeks in Kenya working with AMPATH, and over a month in India working with ARTH.

### *1.8 Summary*

This chapter introduced the components of data collection campaigns and described these components in the context of development scenarios. Furthermore, we discussed the problems often faced by organizations attempting to implement data collection campaigns in developing regions. We then covered early technological solutions based on PDAs and feature phones, their limitations, and lessons learned from the literature about the design and deployment of these systems. Our review of the space provided the basis for why we decided to build a new system and what we hoped to accomplish in doing so. Finally, we presented our solution, Open Data Kit, and introduced its contributions, which will be the focus of this dissertation.

The remainder of this dissertation discusses Open Data Kit, the tradeoffs in the major design decisions, and case studies of several deployments highlighting how the design and implementation of ODK helps mitigate many of the problems noted by organizations (Section 1.2): cost, time, supervision, training, and data quality. Chapter 2 describes the Open Data Kit system, introduces each component, and details how each is used. Chapter 3 further discusses the system's architecture and design decisions related to modularity, demonstrating the benefits of each through real-world examples of modifications and contributions by the community created around ODK.

Chapters 4, 5, and 6 present three detailed case studies of organizations that deployed Open Data Kit for data collection campaigns, and highlights the lessons learned from each. Finally, Chapter 7 summarizes the dissertation, reviews the contributions and lessons learned, and discusses likely productive future directions uncovered by our work.

# Chapter 2

# OPEN DATA KIT

Open Data Kit is a modular set of components that can be composed into a complete solution, used individually for specific tasks, or extended with new modules to create mobile data collection services. The three major components are Build, a form designer, Aggregate, a server for storing blank and completed forms, and Collect, a mobile device client for completing forms. To ensure the interoperability of all the components, we leveraged the XForms standard as their common interface. This chapter discuses XForms, each of the system components, their interaction and use in detail, and describes their operation in a typical data collection deployment. All of the source code for Open Data Kit is freely available under the open source Apache License, Version 2.0 [1], and can be found at: `http://code.google.com/p/opendatakit`.

## *2.1 XForms: A Common Format*

To ensure that each of the tools could be used independently but easily interact with each other, we chose to use the XForm standard [126] as a common data interface. XForms are an XML-based form description standard designed by the World Wide Web Consortium (W3C) for the next generation of web forms. Though XForms have seen little adoption as web pages or forms on the Internet, they have been adopted as a de facto standard for data collection on mobile devices. Already systems such as JavaRosa [73], EpiSurveyor [47], CommCare [30], openXdata [90], and Orbeon [91] use XForms as their form and data standard.

Open Data Kit specifically implements the OpenRosa [89] subset of XForms. OpenRosa is a consortium formed in an effort to create a common set of standards to allow mobile data collection tools to interoperate. Current member organizations contributing to the design and development of OpenRosa's API include: Cell Life [21], Dimagi [42], DataDyne [34], Satellife [106], openXdata [90], and ODK [87]. Rather than implement the complete XForm standard, which is considerably complex and currently consists of nearly 175 pages of documentation, the consortium selected a subset of XForms deemed relevant to the purposes of data collection. Leveraging the OpenRosa standard made our tools backwards compatible with many existing tools commonly used by the NGO community in development scenarios. Furthermore, ODK's OpenRosa compatibility allowed organizations currently using these systems to easily upgrade to different technologies without needing to completely overhaul their entire systems.

XForms were designed to have representations for all aspects of a form including navigation through the form, how the form looks, and how the data is stored. Additionally, XForms provide mechanisms for input validation through constraints, programmatic logic (including data-dependent navigation), and display preferences. These aspects are captured by dividing the form into three distinct sections, the *instance*, the *bindings*, and the *body*. Figure 2.1 shows an example of a basic XForm containing a single question with the three sections highlighted. The instance represents the data storage component, the bindings contain the form logic, and the body specifies the rendering to the user. Each of ODK's tools leverage different portions of an XForm to perform its task. Build creates the XForm, populating each of the three sections with the specified look and logic. Collect uses the body elements to determine how to display the form to the user, the bindings to determine navigation and input constraints, and the instance to determine how to store the data locally. Aggregate uses the instance to determine storage variables and uses the bindings to determine data types when building a database to store the collected data.

```
<?xml version="1.0"?>
<h:html xmlns="http://www.w3.org/2002/xforms" xmlns:h="http://www.w3.org
    /1999/xhtml" xmlns:ev="http://www.w3.org/2001/xml-events" xmlns:xsd="
    http://www.w3.org/2001/XMLSchema" xmlns:jr="http://openrosa.org/javarosa"
    >
  <h:head>
    <h:title>Basic XForm</h:title>
    <model>
      <instance>
        <data id="basic">
          <StringPrompt/>                                    <!-- instance -->
        </data>
      </instance>
      <bind nodeset="/data/StringPrompt" type="string"/>  <!-- bindings -->
    </model>
  </h:head>
  <h:body>
    <input ref="StringPrompt">
      <label>please enter a string</label>                  <!-- body -->
    </input>
  </h:body>
</h:html>
```

Figure 2.1: Example of a basic XForm. The beginning and the end of the Instance section is highlighted in red, the Bindings section is highlighted in blue, and the Body section is highlighted in green.

### *2.1.1 XForms Components*

#### *2.1.1.1 Instance*

The form instance defines the variables representing the data that will be stored. Instance variables can be thought of as column headings in a table. Variables at this stage have no associated data types, and are represented as type-less XML tags. Instance variables may be used to specify default values within an XForm.

The instance is the only section of an XForm that can stand alone – an independent instance is used by XForms as the representation of a completed form. Figure 2.2 shows an example form and a corresponding completed instance. External instances can be read when loading an XForm and used to pre-populate data fields or to re-populate them in the case of loading a partially completed form.

```
<?xml version="1.0"?>
  <h:head>
    <h:title>Sample XForm</h:title>
    <model>
      <instance>
        <data id="sample">
          <name/>
          <sex/>
          <age/>
        </data>
      </instance>
      <bind nodeset=.../>
      ...
    </model>
  </h:head>
  <h:body>
    ...
  </h:body>
</h:html>
```

(a) Sample XForm

```
<?xml version="1.0"?>

  <data id="sample">
    <name>Carl</name>
    <sex>male</sex>
    <age>25</age>
  </data>
```

(b) Completed Instance for Sample XForm

Figure 2.2: Sample XForm and a corresponding *instance* representing a completed form.

#### *2.1.1.2 Bindings*

Bindings contain all of the logic pertaining to what the form does. Here the form designer can specify data types, constraints, branching logic, and any other form or question-level logic. Figure 2.3 shows three example bindings. Answers, constraints, and navigation can be calculated using either pre-defined values or by comparing to one or more previously entered answers in the form. Each binding has a one-to-one mapping to an instance variable using the nodeset attribute, defining the data that can be stored within the given variable. However, instance variables are not required to have an associated binding. Instance variables without bindings can be used as placeholders for values, and if shown to the user, default to a string field.

The representation of branching navigation within XForms differs from that of many standards. Rather than explicitly specify the next question to show based on a given answer, as in a "goto", XForms uses the declarative concept of "relevance", where the decision to show a given prompt is based on previously entered values. However, using relevance means the order of questions is actually defined in the form's body section rather than in the bindings with the question's logic. In the absence of specified relevance logic, the navigation proceeds linearly through the body from top to bottom. When relevance logic is present, the navigation first proceeds top down, with each prompt determining whether or not it should be shown based on the logic in the binding. We found this dually located specification to be very confusing for new form designers.

#### *2.1.1.3 Body*

The form body describes the information to be displayed to the user within each prompt. The body also defines the order that prompts will appear, branching logic notwithstanding. Similar to the binding, body elements must reference one of the

```
1) <bind nodeset="/widgets/select" type="select" constraint="not(selected(., "c") and
     selected(., "d"))" jr:constraintMsg="c and d cannot be selected together"/>

2) <bind nodeset="/widgets/int" type="int" constraint=". &lt; 10" jr:constraintMsg="
    number cannot be above 10"/>

3) <bind nodeset="/widgets/date" type="date" constraint=". &gt;= today()"
    jr:constraintMsg="only future dates allowed"/>
```

Figure 2.3: Example Bindings showing 1) a constraint that a multi-select question not have both option "c" and option "d" selected at the same time. 2) That the current question must have a value less than 10. 3) That the current answer must be a date after the current date.

variables in the instance using the ref attribute, but an instance variable is not required to have a body element. Instance variables without a body element are never explicitly shown to the user, but can be used to store calculated values.

### *2.1.2 OpenRosa XForms Extensions*

In addition to the subset of XForms implemented by the OpenRosa standard, OpenRosa also extends the standard to include other features like dynamic multiple-language support, embedding multimedia into prompts, and preloaders for automatically collecting dynamic information such as dates and times. These features have proven to be incredibly useful for data collection systems, however adding this superset of features breaks compatibility with standard XForms.

#### *2.1.2.1 Preloaders*

Preloaders allow the form designer to specify information to be collected automatically at the time of form entry. This information can include dates and times of form filling sessions, or device information like serial numbers or SIM card identifiers. Preloaders require an instance and a binding, but are never shown to the user.

#### *2.1.2.2 Multi-language Support*

The OpenRosa implementation of XForms introduces the <itext> tag to support dynamic switching of languages within a form as shown in Figure 2.4. Within an <itext> block multiple <translation> tags can be defined, and each language can define the same <text> entries to be shown for a given field when that language is selected. Tags in the body reference these variables using ref="jr:itext=(...)" to specify when a string should be localizable.

#### *2.1.2.3 Multimedia Support*

To add support for embedding multimedia within questions and multi-select choices, we overloaded OpenRosa's itext tag functionality by adding multiple value types to each "text id" tag. An example of the multimedia support is shown in Figure 2.5. Form designers can include audio files that "read" question text or provide example sounds. They can similarly use images to show examples of symptoms or display languages not supported by the Android operating system. Videos can be used to demonstrate how to examine a patient for a specific symptom or shown to a patient for educational purposes. An added benefit of using itext for multimedia is that different multimedia can be defined for each language. This allows a form designer to, for example, include an audio file reading a question in French for a question with English text, and conversely include an English audio file for when French text is displayed. This alternating of languages can help enumerators communicate with clients when they do not share the same language.

```
<?xml version="1.0"?>
  <h:head>
    <h:title>Sample XForm</h:title>
    <model>
    <itext>
      <translation lang="english">
        <text id="name-question">
          <value>What is your name?</value>
        </text>
      </translation>
      <translation lang="spanish">
        <text id="name-question">
        <!-- must escape accents and characters as in ¿Cómo? -->
          <value>%C2%BFC%C3%B3mo se llama?</value>
        </text>
      </translation>
    </itext>
      <instance>
        <data/>
          <name/>
        <data/>
      </instance>
      <bind nodeset="/data/name" type="string">
    </model>
  </h:head>
  <h:body>
    <input ref="/data/name">
      <label ref="jr:itext("name-question")"/>
    </input>
  </h:body>
</h:html>
```

Figure 2.4: itext example

```
<itext>
  <translation lang="english">
    <text id="name-question">
      <value form="long">What is your name?</value>
      <value form="image">jr://images/stoat.png</value>
      <value form="audio">jr://audio/name-eng.wav</value>
      <value form="video">jr://video/name-eng.mpg</value>
    </text>
  </translation>
  ... <!-- other translations -->
</itext>
```

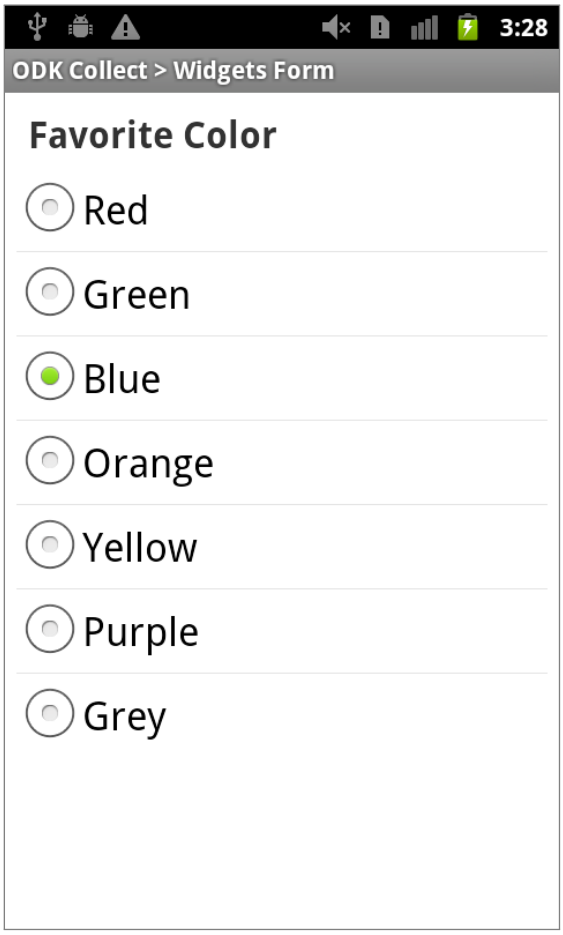
Figure 2.5: media example

#### *2.1.2.4 Appearance*

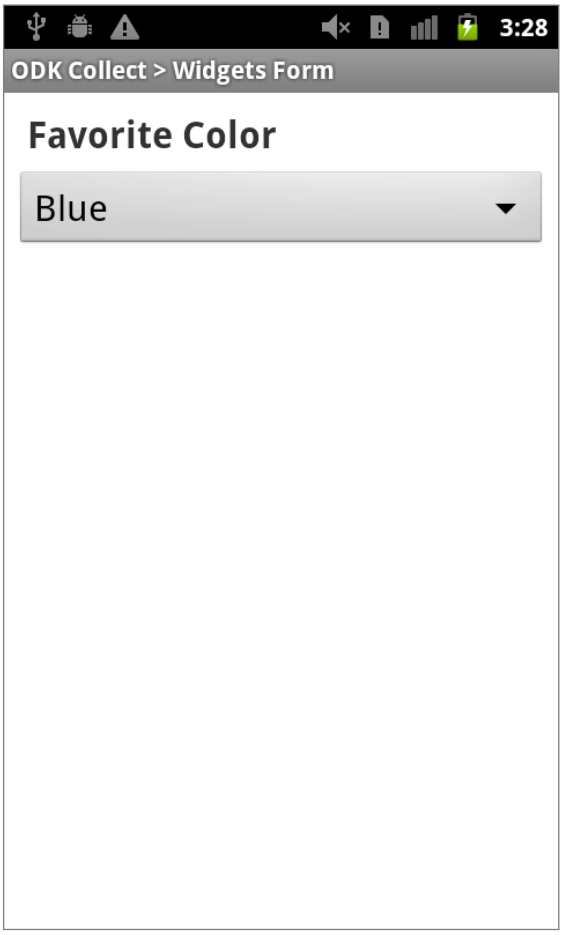
One of the more requested features by users of the system was to customize how certain data types worked. For example, a select-one list defaults to radio buttons as seen in Figure 2.6a, but the same list could be represented as a drop down menu as seen in Figure 2.6b, instead. Similarly, some organizations requested a slider interaction or buttons with plus/minus indicators to select an integer result. To enable this functionality, we created an *appearance* tag in OpenRosa. Form designers could add the appearance tag to the specification of a prompt in the body of the XForm to override the default behavior. The example body element in Figure 2.6c creates the drop down menu shown in Figure 2.6b.

### *2.1.3 Summary*

The feature set provided by XForms made it a good choice by the community as a data collection standard. Above and beyond describing basic form layout, XForms gives authors the ability to define answer constraints, specify form navigation logic, and customize some aspects of the look and feel. The extensions made by OpenRosa further enhanced XForms by adding the ability to define multiple languages and




(a) Select One




(b) Drop Down

```
  ...
<select1 ref=''color'' appearance=''minimal''>
  ...
```

(c) Appearance Example to Use a Dropdown

Figure 2.6: Examples of using the appearance tag

dynamically switch between them while completing a form, as well as the ability to embed multimedia objects within a form.

By defining each of the look, functionality, and data model for a form, XForms allows all of our tools to use the same data structure to complete their given tasks. Having a common data interface for each tool contributes greatly to the system's modularity and ease of use, as the user only has to define one object that all the tools can use. Similarly, by leveraging a common, open standard, components can be easily replaced or new components can be added to the workflow.

## *2.2 ODK: Components*

This section describes each of three components of ODK and their capabilities. We describe each of the components in the order they are typically used in a deployment. A discussion of their use is deferred to Section 2.3

### *2.2.1 Build: Form Designer*

In order to allow non-programmers to build forms, we designed a graphical web-based form designer called ODK Build that allows users to create forms using a drag-and-drop metaphor. By using the XForms standard (discussed in detail earlier in Section 2.1), ODK provides authors with flexible and powerful options for designing forms to fit their needs. Unfortunately, this comes at the cost of ease of use. The structure and syntax of the XML-based format is more complex than is needed to build the majority of forms, which are usually short and linear. This often leads to confusion among users and presents a barrier to adoption.

ODK Build enables users to easily generate XForms without a detailed working knowledge of the XForms standard. Build lets designers drag and drop each prompt the user will interact with onto a canvas as shown in Figure 2.7. Each prompt has a set of properties that users can edit: prompt text, hint text, data type, default value, read-only constraint, required constraint, and answer constraints. If a user enters an

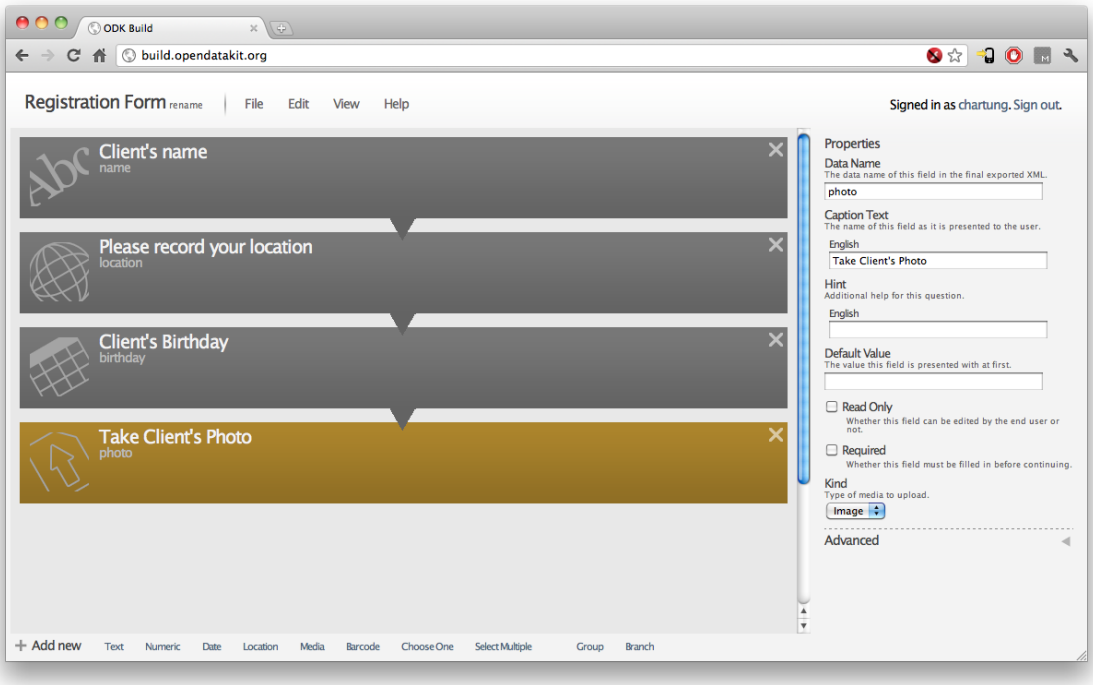



Figure 2.7: In ODK Build, prompts appear on the left of the screen while properties appear on the right. Users rearrange prompts using a drag and drop interaction in the web browser.

illegal value, Build highlights the error in red and will not generate an XForm until the error is corrected. Additionally, Build supports adding multiple languages and custom logic to each question within a form. Users can easily rearrange ordering of prompts through the drag and drop interface.

In keeping with the aim of providing more accessible methods for authoring forms, a number of compromises were made with regards to the more complex but less commonly used features of XForms. For instance, while Build still provides a means to implement more advanced constraints, it highlights only basic range constraints for most data types; and, while in reality XForms defines whether or not to display each prompt based upon a set of rules specific to that prompt, we optimize for the more common case and represent conditional logic as a simple branch, with each branch containing multiple inter-related prompts. Forms requiring more advanced logic can be generated by other tools requiring a higher level of user programming ability such as XLSForm [49].

Build allows users to save multiple forms online and retrieve them later for continued editing. Additionally, Build lets users download forms in an ".odkbuild" format so that the forms can be sent to others, loaded into Build, and edited at a later time. Currently, Build is a one-way tool and does not support the import of already existing XForms.

Once a user has completed his or her form using the drag and drop interface, Build allows the user to view the XML that will be generated to create the XForm and to download the form as an XML file onto their local machine. Once a user has downloaded the XML file from Build, they can continue to manually customize their form if more advanced features are needed, or load their form onto servers or mobile devices.

### *2.2.2 Aggregate: Server Storage*

We designed Aggregate to simplify the process of building and managing a web-based data storage system by creating an application that is usable by people with only basic computer skills. Aggregate comes in a "click-to-install" application package, automating the process of setting up a data collection server through a series of user prompts. Once set up, Aggregate provides a server repository to receive and manage collected data, and it provides standard interfaces to extract data (e.g., spreadsheets, queries, etc.) and integrate with existing systems via HTTP web requests. Aggregate is designed to be a generic data storage service that will run on the user's choice of computing platform. Importantly, it can receive data from any phones or servers that are OpenRosa compliant.

Users simply need to upload an XForm to Aggregate, and it automatically creates the new relations in the data store based on the XForm's instance and bindings. The creation of relations and data mappings at runtime is unlike traditional object relational mappings (e.g., Hibernate [11]), that abstract database details at compile time instead of runtime. Similarly, users are able to construct queries using a web interface without knowing the syntax of the underlying database's query language. Aggregate's web interface also provides a report interface to retrieve data in addition to the ability to export the stored data in common data formats such as CSV, KML, and JSON.

In addition to data management, Aggregate provides four levels of user management for controlling who can manage, upload, and view data. User accounts can be designated as one or more of: Administrator, Form Manager, Data Viewer, and Data Collector. Administrators are automatically granted all three other permissions, however non-administrator users must be granted one or more of the permissions by an administrator. A universal "*Anonymous*" account always exists and can be set to have any permission level except Administrator. If "*Anonymous*" is not granted

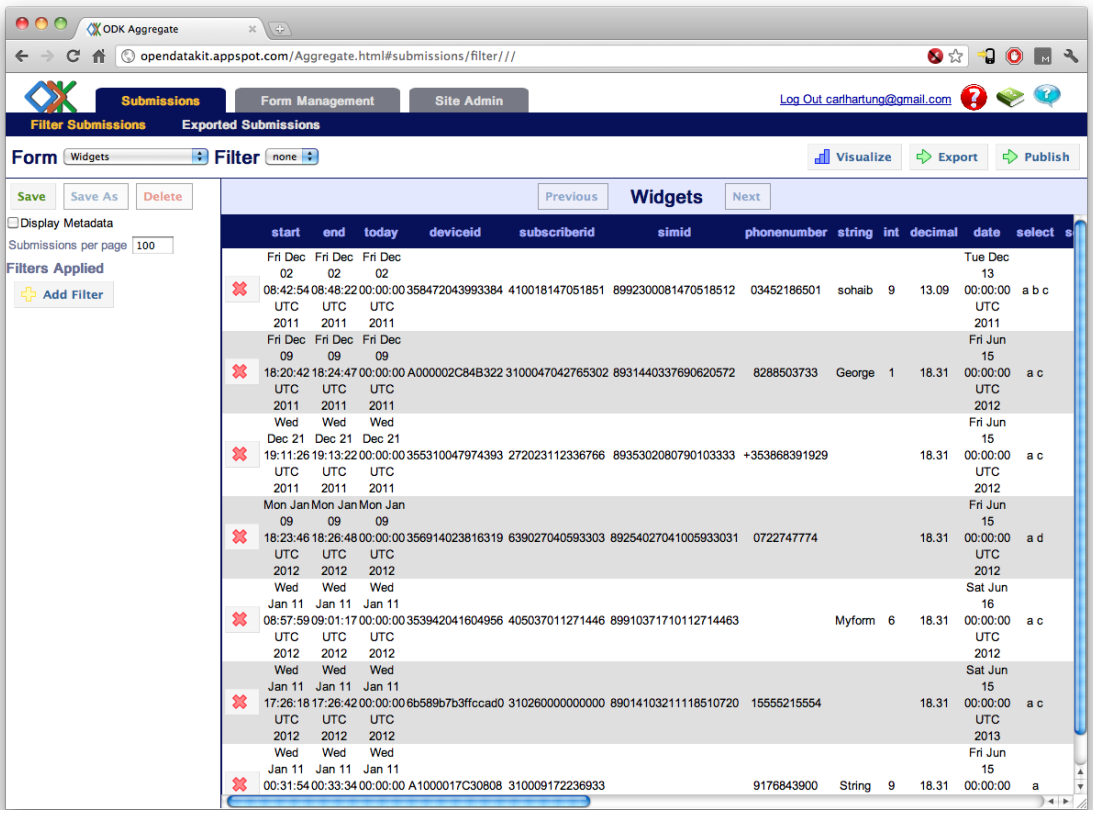



Figure 2.8: ODK Aggregate submissions in its default table format. Each row is a completed form, and each column represents the data variable from each form.

a permission such as Data Collector, then all users attempting actions for that permission (in this case, attempting to submit data to the server) will be required to authenticate themselves using a username and password. Having these levels of access helps to prevent the insertion of malicious data, protects the privacy of the data collected, and prevents unauthorized modifications to forms.

Once an XForm definition has been uploaded to an Aggregate server, data collectors can begin sending completed instances to the server. Data residing on the server is presented to the user in table format as seen in Figure 2.8. Aggregate is meant to be the data store "*of record*", and accordingly records in Aggregate cannot be modified. However, users with appropriate access have the ability to delete entire records from the data store.

Rather than allowing administrators to directly edit data in Aggregate, we provide several mechanisms to export the data in formats that can be read and manipulated by other programs. Additionally, Aggregate provides an API for streaming data directly to other web services that we term "publish". Aggregate provides publishing to Google's Fusion Tables and Spreadsheets by default, but other services may be added using the publish API. The API allows sending of all data currently in the database, only future data, or both. If future data is used, the data is pushed to the selected service immediately when Aggregate receives the data.

### *2.2.3 Collect: Smart Phone Client*

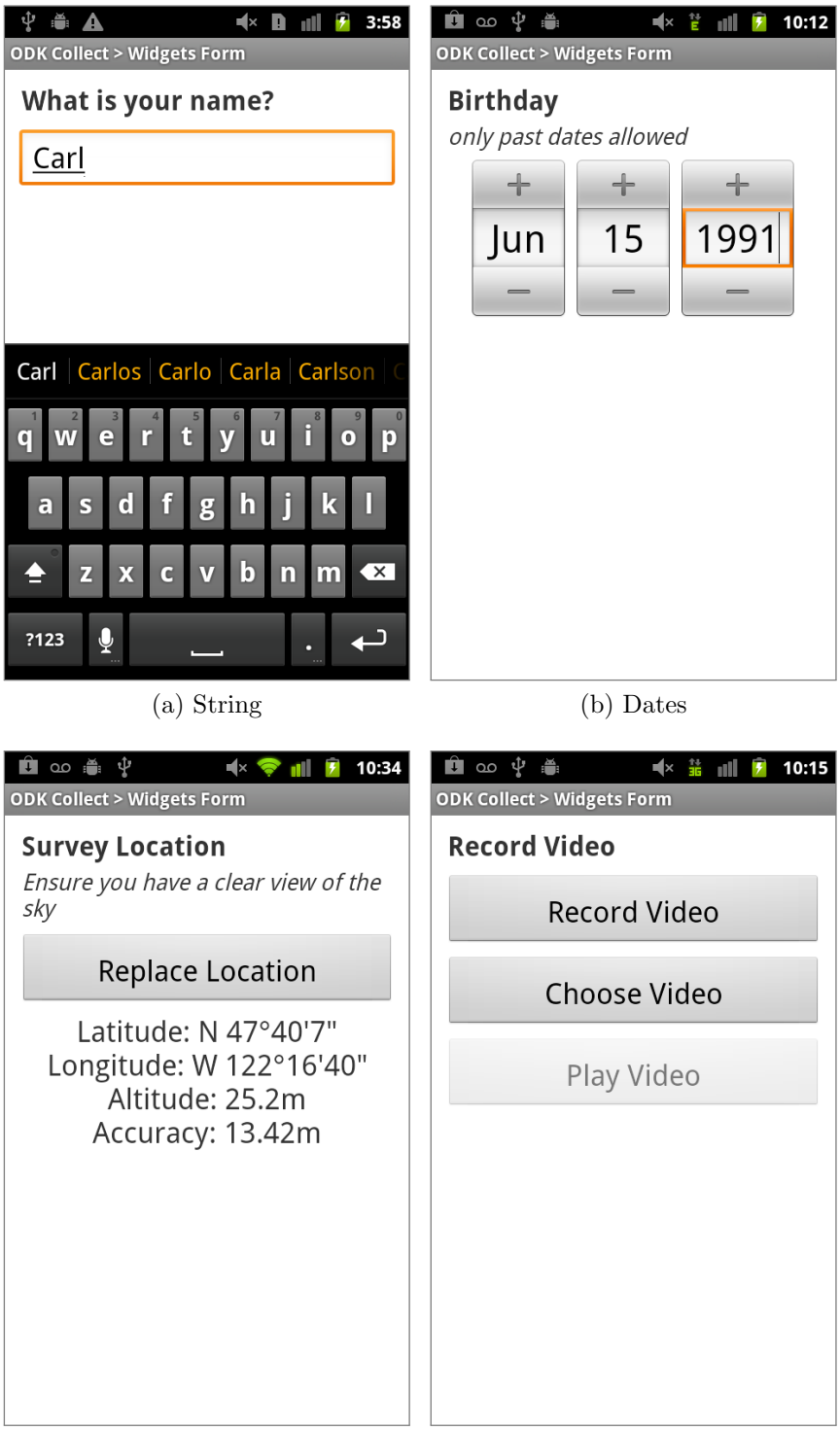
To provide field workers an easy way to complete forms we created Collect, a client for mobile smart phones that run the Android operating system. Collect was designed to be used by workers with little to no formal education, who may have never used a smart phone before. Wherever possible we employ natural gesture-based interaction, provide the ability to include multimedia to guide illiterate users, and constrain data input to catch and correct errors at the time of entry. Collect's goal is to make the entire process of collecting data in the field faster, easier, and more accurate.

To facilitate data collection, Collect uses the XForm's *bindings* and *body* to render prompts to the user to collect or convey information. Prompts can be specified as one of the following: free text, integer, decimal, date, time, select-one (radio buttons), select-multi (checkboxes), image, audio, video, barcode, or GPS location. Any prompt element can also be made read-only to communicate, rather than record, information. For each type of prompt, Collect supports real-time input constraint checking, including regular expression matching of entered data. Constraints may be specified in terms of constants, such as *not more than 4*, or they may be relative to results from other prompts, such as *equal to the sum of prompts X and Y*. Additionally, prompts can be specified as "*required*", preventing the user from advancing without first recording an answer to the prompt. Figure 2.9 shows common examples of prompts and messages within ODK.

By default Collect renders a single prompt on the screen at a time. However, Collect also supports the grouping of prompts and can be configured to display multiple prompts on the same screen within a given group. For data that may repeat in a form, Collect supports "*repeat*" functionality, displaying the same group of questions more than once. For example, a socio-economic survey may want to record demographic information (age, sex, etc...) about all members living within a single household in addition to general information about the household itself.

As the user proceeds through the form, Collect automatically calculates branching conditions to render questions in the proper order. For example, in a medical form, a question asking for the sex of an individual would likely be followed by a question about being pregnant if "Female" is selected for the first answer. Answering "Male" to the initial question would skip the pregnancy question and move on to the next prompt.

To navigate forward and backwards through forms, users interact with Collect by using a "swipe" motion. This motion is akin to turning a page in a book, a motion familiar to almost all users. On the occasion that a user would like to return to




(a) String (b) Dates

(c) Location (d) Video

Figure 2.9: Examples of common widgets in Collect

a question much earlier or later in the form, Collect supports a "jump to" feature, displaying all the relevant questions for the current session.

Question prompts and elements of select-lists can contain images, videos, and/or audio in addition to or in place of text. Examples of multimedia use in ODK deployments include organizations using videos for educating clients, as well as reminding data collectors of processes. Another organization used images of animals to help data collectors properly identify different species. Furthermore, we have used images of scripts in place of text to support languages not included in early versions of Android. For languages included in the unicode standard, Collect supports dynamically switching text among multiple languages.

Since field workers often gather data from areas with intermittent or no Internet connectivity, we designed Collect to support disconnected operation by storing all forms and resulting data on the phone as XML files along with associated binary files (images, audio, video, etc...). Partially completed forms can be saved locally on the phone and accessed at a later time to be completed or edited. The user can choose to synchronize with a server at any time using any available internet connection. Files are sent and received by directing standard HTTP POST and GET commands to any OpenRosa compatible server. Files may also be transferred to and from the phone with a USB cable connected to a computer, or by removing the phone's SD card and accessing its contents using another device such as a laptop.

## *2.3 Operation of the Toolkit Components*

This section discusses how an organization would operate the tools during a typical deployment. More detailed instructions can be found at `http://opendatakit.org`

The intended overall workflow of the components is to:

1. Create the form to be used by the field workers. (*Build*)

2. Set up the server where the data will be stored. (*Aggregate*)

3. Upload the form to the server. This will both A) build the database where the data will be stored and B) provide a common location where the mobile clients can retrieve the form. (*Aggregate*)

4. Download the form onto the mobile clients. (*Collect*)

5. Use the mobile clients to gather and upload data. (*Collect*)

Figure 2.10 shows how each of the system components fits together.

### *2.3.1 Step 1: Build a Form*

The first step in the data collection process is to build the form used to collect data. We created ODK Build for this functionality. To make Build widely available, we host a running instance on our own server found at `http://build.opendatakit.org` so that anyone with a web browser on the internet can immediately begin authoring forms for their own use and access them from anywhere. Build is free for anyone to use. More intrepid developers can download the source, built using Ruby Rack, and host their own instance on their own machines.

After navigating a web browser to `http://build.opendatakit.org`, the user is presented with username and password fields. An account is not required to create and export forms in Build, and the user can simply hit "cancel" to continue. However, if a user wishes to save a form in Build and edit it later, an account is required.

After logging into the site, the user can author a form, creating and ordering prompts as desired. After completing their form, the user can choose "Export to XML" in order to see the raw XForm that will be produced. This screen also presents an option to download the form to the user's local machine. If the user wants to edit the form later, they must save it within Build. Users may also download the form in an .odkbuild format that can be imported into Build later for sharing with other users. Build does not yet support importing raw XForm files.

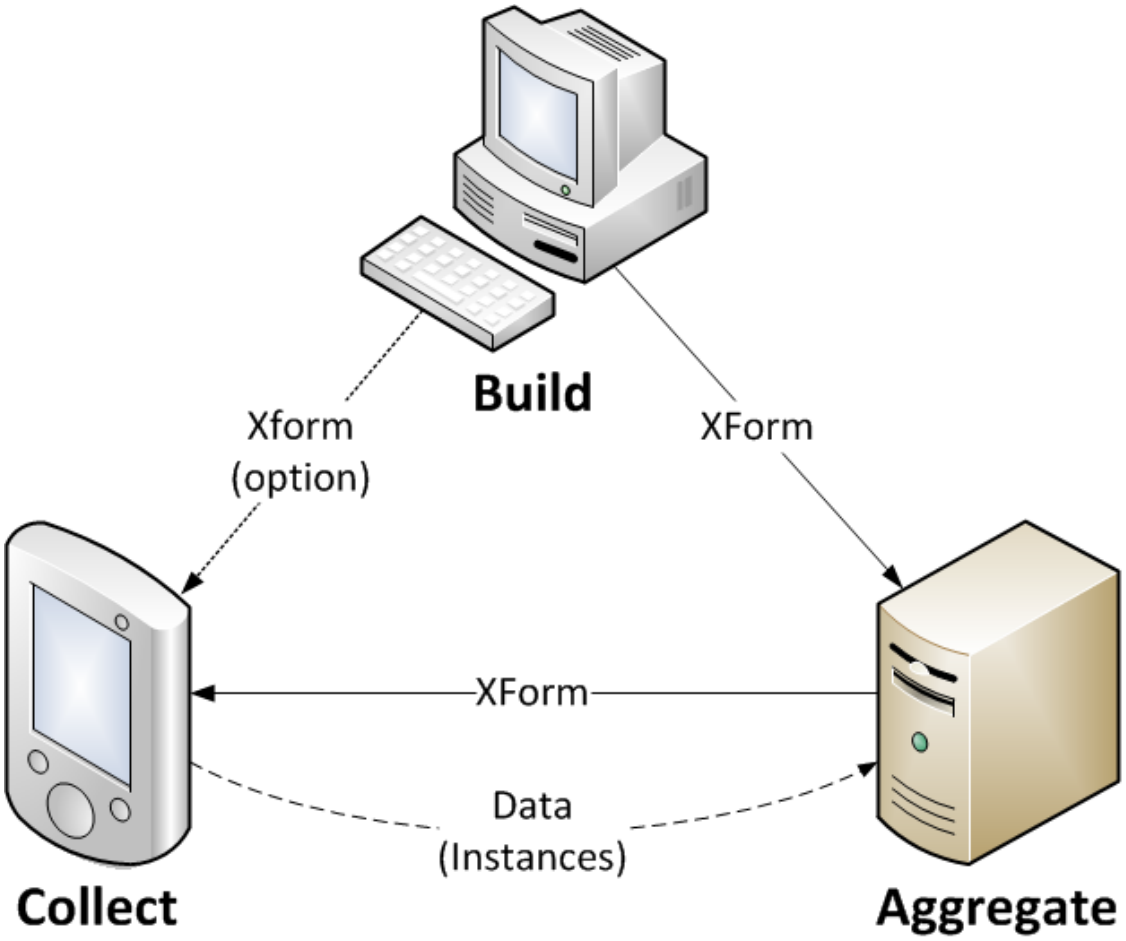



Figure 2.10: The current components of ODK. Build, a form designer that generates XForms containing the logic for user interaction and for data store creation. Collect, a client that renders the form on a mobile and sends entered data to Aggregate, the server for data storage.

Once the user has downloaded the .xml file representing their XForm, they are ready to create a server.

### *2.3.2 Step 2: Creating a Server*

When creating a server using Aggregate for their deployment, the user must first decide whether to host the server on a local computer or whether to host it on a cloud infrastructure. If hosting locally, the user must first install Tomcat and a database, one of MySQL or PostgreSQL. If hosting in the cloud, the user must create an account and instance on Google's AppEngine. Once installed on either platform, the operation is the same.

We provide three separate installers for each of the major operating systems: Windows, Linux, and OSX. The choice of installer relates only to the operating system used to run the installer. During the installation process the user can choose where to install Aggregate, that is, locally or in the cloud. After completing the installation process, a functional Aggregate server will be running and ready to use.

After logging onto the server for the first time, the user can configure the desired access and security settings for their deployment. Next, the user can upload the .xml file created by Build in step 1 by selecting "Add New Form". Once the form is uploaded, Aggregate automatically builds a database specific to that form. There is no limit to the number of forms that may be uploaded to a given Aggregate instance.

After the form has been uploaded and the datastore for the form is created, the server is ready to begin receiving data.

### *2.3.3 Step 3: Configuring the Mobile Client*

The first step in configuring the phone is to install Collect. Collect is compatible with any phone running the Android operating system version 1.6 and higher. Collect can either be obtained freely from Google's Market (recently renamed Play Store in some

regions), which comes installed on most phones, or it can be downloaded and installed by navigating to the .apk file on our website with the phone's web browser.

After installation, the user must configure the phone to communicate with Aggregate by entering the server's URL into the server preferences option. The user may then add the form to the phone's SD card in one of two ways. By selecting "Get New Forms" from the main menu, Collect will communicate with Aggregate and display a list of available forms that have been uploaded to that instance. The user can choose one or more forms to download. Alternatively, the phone can be connected to a computer with a USB cable, the SD card mounted to the local machine, and the .xml file can be copied into "/sdcard/odk/forms/".

This process needs to be repeated for all phones involved in the campaign.

### *2.3.4 Step 4: Collecting Data*

Once the phone has been configured to communicate with the corresponding Aggregate instance and the form or forms have been downloaded or copied onto the phone, pressing "Fill Blank Form" will present a list of available forms ready to be completed.

When completing a form, instances may be saved as "Finalized" or "Incomplete". Incomplete instances appear when the user selects "Load Saved Form" and can be completed at a later time. A user marks instances as finalized when each instance is complete and requires no further edits.

After completing one or more instances of the forms, the user sends the data to the server by selecting "Send Data". This presents a list of finalized instances, allowing the user to select one or more instances to upload. After attempting to send the data, Collect notifies the user of the success or failure of each instance.

### *2.3.5 Step 5: Analyzing the Data*

When users begin submitting data to Aggregate, the data will appear in a table-format under the Submissions tab. Aggregate provides mechanisms to do simple

visualizations like pie charts and bar graphics when selecting the "Visualize" button. Users may also export the data in .csv or .kml files for analyzing with statistical and geographical packages, respectively. Alternatively, users may choose to "Publish" the data, streaming it to another web service.

At any point in the process users may create new forms, upload them to Aggregate, download them to the phones, and begin collecting new data. This is the typical use case envisioned by ODK.

## *2.4 Summary*

This chapter has presented the components of Open Data Kit: Build, Aggregate, and Collect. First, we discussed the data interface we choose to allow interoperability between the components, XForms. Next, we discussed the functionality of each tool and the services they provide. Finally, we discussed a typical scenario for deploying the tools and the roles of each tool within that scenario.

# Chapter 3

# EXTENSIBILITY AND MODULARITY

By creating a system with independent components, rather than using a tightly coupled, monolithic approach, ODK has enabled organizations to take a best-of-breed approach to building complete systems for their particular campaigns. This ability to wholly replace each module within the system has proven very important, and empowered groups to make different modifications and additions to fit their needs outside of the scope of ODK's original vision. In addition to creating the system in a modular fashion, we specifically focused on making the design of Collect modular, separating out functionality like how the data is stored locally on the phone and how it is sent to data storage servers. Because of this, we have found that most organizations choose to use Collect as their mobile client, with one or two of its internal modules replaced, and integrate it with their own existing infrastructure. The ability to create and modify the system has fostered a community approach to building, extending, and adding new features to data collection systems. This chapter discusses some of the key design decisions relating to ODK's modularity and extensibility, and presents a variety of new modifications and integrations by the community made possible by our design choices.

## *3.1 Form Design Modules*

We created ODK Build specifically to target novice users building simple forms. However, in doing so we obscured some of the more advanced features of XForms. Often there are tradeoffs between providing functionality and ease of us, and we opted to design Build for ease of use to make getting started with the system simpler.

To satisfy more advanced users and scenarios, several organizations have stepped in and built their own form designers, each providing a different level of sophistication and interface. However, each also creates the same XForms as Build. Dimagi has developed Vellum, a web-based form designer similar to Build, though its interface is a little more open-ended, relying on the user's knowledge of XForms to create the appropriate statements. EpiSurveyor, discussed more in Section 3.4 has its own online form designer as well. Finally, XLSForm, designed in collaboration with the Modi lab at Columbia University and the ODK team at the University of Washington, took a different approach to form design. Rather than provide a graphical user interface, users can design a specially formatted Excel spreadsheet where each row represents a new prompt, and each column represents attributes relative to the prompt. In this way XLSForm mimics a programming language to create XForms. While not necessarily as simple as Build, most advanced users seem to prefer XLSForm as it allows unfettered access to, and simplifies the creation of, the more advanced XForm features. Once complete, the .xls file can then be converted to an XForm using the XLSForm compiler that is implemented using Python.

Though Build has proven to be an important avenue, giving new users an easy way to begin using the system, the ability to replace Build with the other systems listed in this section gives more sophisticated users access to the advanced features of XForms.

### *3.2 Server Modules*

We built Aggregate to be as flexible and extensible as possible, taking the approach that organizations should be able to manage their own data. Rather than implement a centralized server operated by us, incidentally representing a single point of failure that would affect many users, we provide organizations a choice of platforms and ability to run on cloud infrastructures or on local machines. Additionally, we purposely made the data within Aggregate immutable, so it could serve as the data-store

of record. However, to make the data available, we provide export capabilities in .csv and .kml. Additionally, we made Aggregate extensible through its publish API, allowing organizations to extend Aggregate's functionality by streaming the data to any other web service. In this way, Aggregate can serve as a gateway for the data coming from the phones, forwarding it to the specified services.

Aggregate users can choose to either host their server in Google's cloud infrastructure, or on local machines running Tomcat and either MySQL or PostgreSQL. We give users this option as there are many important tradeoffs with each system. For example, running a system locally requires the IT overhead of owning and maintaining local web-connected servers in addition to costs of electricity, security, and maintenance. With a local system the organization is responsible for creating and storing backups and protecting the system from viruses. Conversely, cloud-systems eliminate these needs as the hosting organization handles security and backups. However, when operating in the cloud there is no way of knowing exactly where data is stored or how many copies exist. Thus, organizations managing sensitive private data such as medical information often do not feel comfortable with cloud systems and prefer that the data remains local and under their control. To meet the various needs of different organizations, we provide the ability to choose either option.

Other organizations replaced Aggregate with centralized data stores to further simplify the process by not requiring organizations to setup and host their own servers. For example, formhub provides a centralized hosting service, seeking to make the sharing of data easier. Other commercial services like EpiSurveyor provide data hosting on the Internet for a fee.

Like Build, the ability to replace the server module with another to fit the needs of the organization has proven to be an important design point. Depending on their needs and capabilities, data storage systems can be managed by the organization locally, managed by the organization but hosted in the cloud, or centrally hosted on the Internet by other organizations.

## *3.3 Mobile Client Modules*

So far two classes of mobile devices can be used with our system: J2ME feature phones running the JavaRosa platform or variants, and Android devices running ODK Collect or variants. We note, however, that organizations have begun creating clients for Blackberry and iOS. Several aspects about Android made it a very promising platform for a data collection system. First, it was the first mobile operating system to give unrestricted programatic access to all of the phone's capabilities, including access to onboard sensors. Also, its Intent system allows for an application to use other applications to do specific tasks, essentially using the second application as a module within the first. Though designed with the same modularity in mind as Build and Aggregate, we built Collect as a modular set of Android components, separating out functionality such as data storage and submission. The modularity within Collect has proven most important as most organizations have chosen to use Collect as their mobile client, replacing one or two of its internal modules, and integrating it with their own existing infrastructure. This section discusses the modularity within Collect that made these integrations possible.

### *3.3.1 Intents and Content Providers*

#### *3.3.1.1 Intents*

Going a step beyond customizing how a certain widget looks and acts, Android gives developers the power to actually execute other programs to do work and return the result. This system is referred to as Android's Intent system. Intents allow for run-time communication between components within or across applications. The idea of an Intent is to specify a desired action along with optionally specified data to act upon. When an Intent is fired by an application, the Android operating system checks for applications and/or components of applications registered to handle the given action and data type. If one exists, that application is launched. If more than

```
/*  take a picture */
// Set the Intent to capture an image
Intent i = new Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
// Start the activity, providing the Intent and the data to return
startActivityForResult(i, FormEntryActivity.IMAGE_CAPTURE);

/* get a picture from a gallery */
// Set the Intent to get content
Intent i = new Intent(Intent.ACTION_GET_CONTENT);
// Set the content to get to image
i.setType("image/*");
// Start the activity, providing the Intent and data to return
startActivityForResult(i, FormEntryActivity.IMAGE_CHOOSER);
```

Figure 3.1: Example Intents to take a picture and to select a picture from a gallery.

one exists, the user is given a choice of which application to execute. If none exists, the system reports an error, signaling the user to install an appropriate application.

For ODK Collect, we leverage the Intent system to invoke all of the media capture applications already included with Android for photos, video, and audio. Two example Intents demonstrating taking a picture and choosing a picture from the gallery are shown in Figure 3.1. Furthermore, the Intent system allowed us to add barcode reading as a basic prompt type without having to implement our own barcode scanner. Rather, users can install any barcode scanning application that responds to the appropriate Intents.

We similarly leverage Intents to allow other applications to use Collect as a data entry service by instrumenting it to respond to external Intents requesting "*VIEW*"ing or "*EDIT*"ing of ODK forms. In this way, external applications can use an Intent to launch Collect, the user can use Collect to enter data into a form, and then Collect returns the path of the newly completed instance to the calling application via ODK's instance Content Provider. The calling application can then use the entered data accordingly. Thus, Collect can act as a module within other applications to perform the task of form filling.

#### *3.3.1.2 Content Providers*

The Content Provider abstraction is Android's primary interface for applications to interact with data. The interface specifies a set of actions that every Content Provider must support, regardless of the underlying data structures. The data underlying a content provider can be of any type, including databases or files. In the case of ODK Collect, we use an SQLite3 database to maintain metadata about form and instance files. However, for enhanced modularity, almost all actions are performed on files themselves, and we use background threads to update the Content Provider's database to reflect the contents of the disk. In this way, other applications are free to use the Content Provider to access the forms and data, but the user or other applications can still manipulate the files manually.

### *3.3.2 Decoupled Communication*

For convenience, Collect provides built-in submission and form download mechanisms using standard HTTP POST and GET commands. However, with the Content Provider and file information freely available, a developer can either disable the HTTP communication or replace it with other applications. We left the communication component decoupled to allow organizations to develop transmission modules using other mechanisms such as SMS. Additionally, users can simply connect the phones to computers and copy files to and from the filesystem rather than use wireless communication. We discuss how several organizations utilized this modularity to connect Collect to their own custom backend servers in the next section.

## ***3.4 Community Modification and Customization***

The extensibility and modularity designed into ODK has enabled a wide variety of extensions and modifications created by many organizations to fit their needs. This section discusses the organizations, the modifications made by each, why the organi-

zations felt they needed to create the modifications, and how ODK's design enabled those changes.

### *3.4.1 Grameen AppLab Integration with SalesForce*

Grameen Application Laboratory [57] is an initiative of the Grameen Foundation that works to promote innovation in the provision of services and information using mobile phones and other ICTs (Information and Communication Technologies) to alleviate poverty in the developing world, and was the first organization to deploy ODK in a developing region. We discuss that initial deployment in detail in Chapter 4. However, since that initial deployment several projects within Grameen have adopted parts of ODK for their data collection campaigns. In a particular, they replaced Aggregate with SalesForce as their data repository. Grameen uses SalesForce to "track relationships and manage programs and organizational knowledge" through its advanced data analysis tools, reporting, and a public data interface where anyone can view the progress of their campaign. Figure 3.2 shows an example of Grameen's CKW project on SalesForce. Grameen continues to use Collect as their mobile client, but modified the submission protocol to submit directly into SalesForce. Leveraging the decoupled submission architecture allowing them the important ability to directly integrate mobile data collection into their existing processes.

### *3.4.2 AMPATH and ODK Clinic*

AMPATH [4] is the one of the largest HIV treatment programs in sub-Saharan Africa, and we discuss their use of ODK with community health workers in great detail later in Chapter 5. In addition to that deployment, however, AMPATH has similarly integrated ODK into their clinical workflow. They created a derivative application, ODK Clinic, which can pull patient records onto the form for summary viewing and updating [8]. As a component of Clinic, doctors can select a patient and select a specific form to complete for that patient. Selecting the form opens ODK Collect and

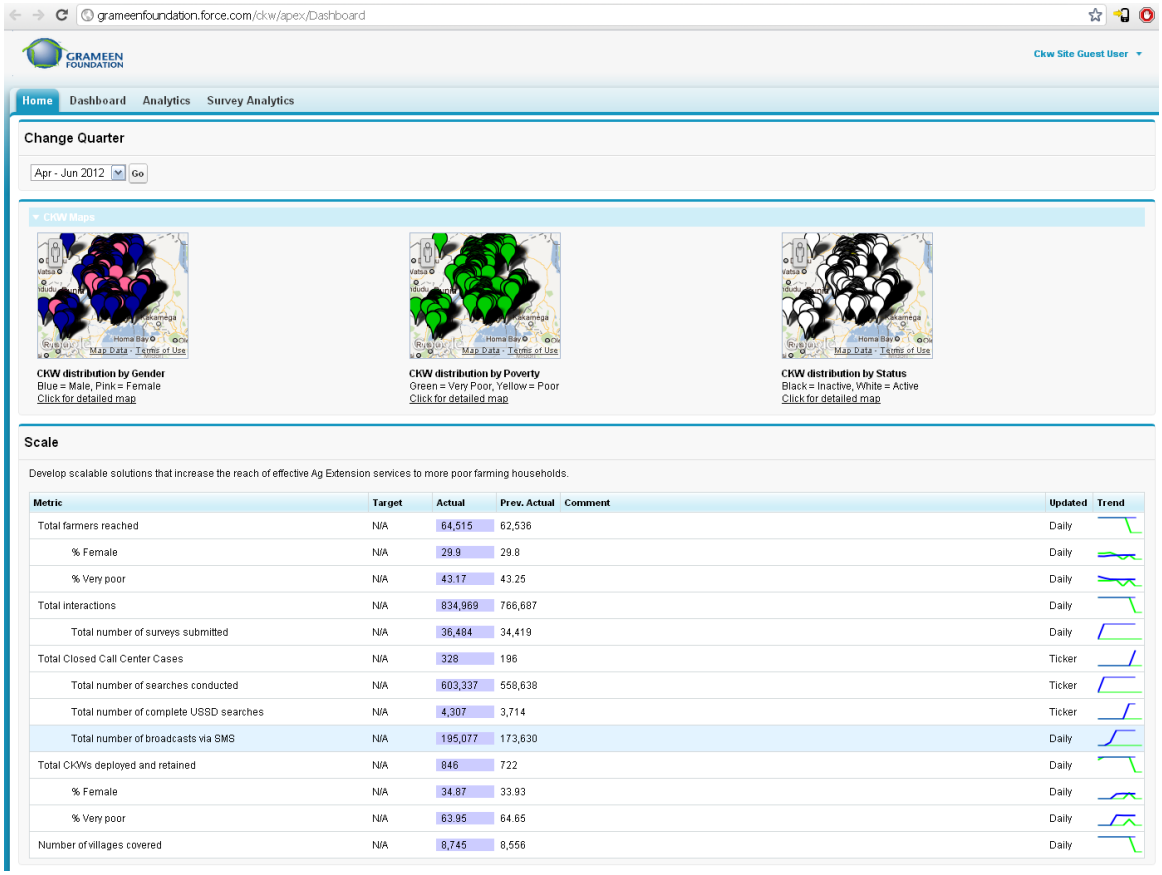



Figure 3.2: Grameen's integration of ODK with SalesForce

pre-populates the form with the patient's known information. Once the form is complete, Collect returns the completed form to Clinic, where the user can then upload it to OpenMRS. In this case, AMPATH used ODK's extensibility to incorporate it as a component of Clinic, much like taking a photograph is a component of Collect.

### *3.4.3 Group Complete and CouchDB*

GroupComplete [58] is a for-profit company offering mobile data collection tools to organizations for a monthly fee. GroupComplete uses ODK Collect as their mobile client, but has replaced the server component with CouchDB [31]. CouchDB is a database system designed for easy replication. Rather than use SQL and a relational database, CouchDB uses JSON to store data where each database is an independent document maintaining its own data and self-contained schema. Additionally, CouchDB uses JavaScript as its query language. Similar to Grameen, GroupComplete took advantage of the decoupled submissions and replaced it with CouchDB to provide seamless form and data synchronization across phones, a feature not natively provided by either Collect or Aggregate. In this way, administrators could more easily control the forms that were available, and all workers had access to all data collected on all phones, enabling real-time collaborative data collection.

Additionally, GroupComplete created a form design application for the mobile phone, allowing the entire process of form creation, data collection, and data analysis to happen only using Android devices; a use case we had not envisioned with our initial design.

### *3.4.4 Modi Lab at Columbia University and formhub*

The Modi lab was established to use engineering to help address development issues. Towards these efforts they created formhub as a centralized data repository to make the data collection process easy and collaborative. Rather than requiring users to set up their own server with Aggregate, Modi hosts a single, web accessible instance

of formhub, allowing users to store and share their forms and data online. Though formhub uses XForms, the Modi lab team decided to hide the details from users. To do so they built a tool called XLSForm to convert specially formatted Microsoft Excel files (.xls) into XForms. Users author and save forms in the .xls format and upload them to formhub, never needing to see the underlying XML.

Modi lab took advantage of ODKs modularity by building a single tool that encapsulates and completely replaces both Build and Aggregate. They still use Collect as their recommended mobile client, which required no modifications to enable the submission of data to formhub. To download forms and upload data, users need to simply enter the URL provided by formhub into Collect.

### *3.4.5 Johns Hopkins Center for Clinical Global Health Education and eMOCHA*

Johns Hopkins Center for Clinical Global Health Education aims to build medical expertise in resource-limited settings. Before ODK Collect supported media embedded within forms, researchers at Johns Hopkins leveraged Android's Intent infrastructure to provide a similar experience by building eMOCHA, a modified version of ODK Collect. From their website, "eMOCHA is designed to assist health programs in developing countries improve provider communication and education, as well as patient care, by coordinating wireless devices with local server-based clinical training and patient care support services." Initially, eMOCHA allowed users to play videos, and at the end of those videos would automatically launch ODK via an Intent with a pre-specified quiz with questions about the video for training purposes. Later, we were able to re-use some of this code in Collect when adding the feature to embed multimedia within forms. The eMOCHA project expanded on ODK further, leveraging its modularity by implementing their own data storage system built with MySQL and PHP on the Kohana framework to replace Aggregate.

By leveraging ODK, eMOCHA provides a system to accomplish the following key steps in their clinical training program: [44]

- To use clinical data collected from the point-of-care, to identify which health care workers in the community need to be trained.

- To analyze the clinical data and identify the most critical knowledge or skill needed.

- To create and efficiently deliver high-quality training content to target these specific training needs.

- To evaluate the impact of this training, based on data collected directly from the field, to ensure that provider knowledge and skills improve and result in better health for their patients.

- To empower the providers in their communities, at the point-of-care, with the tools to monitor and improve their own clinical care practice.

### *3.4.6 DataDyne and EpiSurveyor*

DataDyne's EpiSurveyor platform was one of the first data collection platforms pioneering the use of J2ME enabled feature phones and the OpenRosa standard. Their service provides a centralized repository for forms and data collected with mobile phones. Rather than taking the time and effort to build a new Android client for their system, DataDyne simply modified ODK Collect for that purpose. By replacing the submission mechanism to communicate with the same protocol as their server (which is proprietary), they were able to release an Android client in a very short time and with very little effort.

### *3.4.7 Dimagi, CommCare, and DataHQ*

Another early adopter of the OpenRosa standard was Dimagi in their development of JavaRosa for J2ME enabled phones. Dimagi has leveraged JavaRosa in creat-

ing CommCare, a job aid for community health workers. CommCare provides a multi-level cloud-based storage system for managing patients and forms called CommCareHQ. Dimagi has leveraged ODK Collect in porting the mobile portion of their CommCare platform from J2ME phones to more capable Android phones. Similarly they developed their own generic data storage system in DataHQ, and have already paired it with Collect to conduct a three month, 400 question survey in Mozambique where they collected over 15,000 submissions.

### *3.4.8 VillageReach and OpenLMIS*

VillageReach works to improve access to healthcare for remote, underserved communities around the world. Primarily, they work with partners to improve the performance of health systems in inaccessible and isolated "last mile" communities. They operate a supply chain and cold chain management system using OpenLMIS, a logistics and supply chain management application. OpenLMIS provides a high degree of flexibility with multiple paths for data entry including SMS, Excel, and web forms. By customizing ODK Collect's submission infrastructure to integrate with OpenLMIS, they have extend their reach and enabled real-time in-field tracking of vaccines, medicines, and equipment supplied to rural health clinics.

### *3.4.9 Vestergaard Frandsen, Manna Energy, and the Carbon for Water Project*

Vestergaard Frandsen, working with Manna Energy, created the Carbon for Water Project to provide sustainable access to safe drinking water using carbon-financing. VF used all three components of ODK to collect data about users as they distributed water filters throughout Kenya. As the largest ODK deployment that we know of to date, VF deployed over 4,000 phones with community workers and collected over 40,000 records a day, totaling over 1 million records in a single month. They used the flexibility of Aggregate to send this data into their own reporting systems so that auditors could track the progress of the workers live on the Internet.

### *3.4.10 Google Earth Outreach*

The Google Earth Outreach team leveraged ODK Collect's modularity by integrating it with Google Earth Engine to add an on-the-ground mobile data collection component to their satellite imagery. Google Earth Engine presents the world's satellite imagery along with trillions of scientific measurements over the last 25 years to map trends on the Earth's surface. With a particular focus on monitoring deforestation, they have worked with a variety of groups and organizations around the world such as the Surui tribe in Brazil and the Jane Goodall Foundation in Tanzania.

### *3.4.11 Summary*

All of the organizations and projects listed in this section had specific needs not immediately addressed by ODK. However, rather than force these organizations to start from scratch, ODK's modular design allowed these organizations to customize different pieces, allowing for integration directly into their current workflows.

## *3.5 Community Support*

In order to support the growing ODK community we provide a website with reference implementations, sample videos, an email-list for developers, and an email-list for implementors. We also use Google Code to host the source code, the various releases, and track bugs within the project. We developed training materials from our initial experiences, but now eleven different organizations have posted training materials and tips learned from their deployment experiences to help others get started.

## *3.6 Summary*

For a large number of organizations, integrating mobile data collection is a very desirable, yet sometimes daunting task. Most already run their own data storage, data entry, and reporting systems, and wholesale upgrades can be time consuming,

expensive, and difficult. By providing a modular design with flexible interfaces, ODK enables organizations to integrate as much or little of the platform into their existing infrastructure as necessary. This important ability has created a large community around ODK with members providing advice, experiences, and even source code.

# Chapter 4

# INITIAL DEPLOYMENT EXAMINING FEASIBILITY AND TRAINING

*GRAMEEN APPLICATION LABORATORY, UGANDA*

We partnered with the Grameen Application Laboratory (Grameen AppLab) to do the first developing world trial deployment of ODK. We used this deployment to determine the feasibility of deploying ODK Collect on smart phone devices for data collection campaigns in developing regions, get feedback from Grameen about current features and desired functionality, and to help determine what level of training is required to use the system. As this deployment was exploratory, we had yet to fully implement all of the features of ODK. Instead, we provided a subset of current features for testing and used our experiences with Grameen to guide the direction of the project. To our knowledge this was the first deployment of a data collection system utilizing smart phones in Africa, and certainly the first using Android devices.

After evaluating the deployment Grameen felt that ODK was an improvement over their previous paper system, and the benefits provided by ODK convinced them to switch from their paper system permanently. In addition, we left armed with a set of features requested by Grameen that steered our development towards using XForms to provide a better user experience and improve the quality of collected data. We learned some unexpected lessons about user interface design challenges relating to users in low-resource regions, and discovered that only one to two days of formal training was required, even for users who had never previously used a smart phone.

## *4.1 Background*

The Grameen Application Laboratory [57] is an initiative of the Grameen Foundation that works to promote innovation in the provision of services and information using mobile phones and other ICTs (Information and Communication Technologies) to alleviate poverty in the developing world. During the initial development of ODK Collect, Grameen began piloting an SMS question and answer service for agricultural and health information in partnership with Google in Uganda and wanted to survey users about their experiences with the system.

### *4.1.1 Current Processes*

The SMS question and answer service provided several phone numbers that anyone could send SMS messages to with questions about agriculture, health, or weather. The service would then use a matching algorithm to pick an appropriate pre-defined response based on the question asked, and then return that message via SMS to the asker's phone. The information was primarily targeted at low-income, rural, subsistence farmers who would otherwise have few channels available to them to learn the information. In order to pilot the system and gather feedback, Grameen employed a network of shared phone operator partners throughout the country.

Shared phone operators (SPOs) are typically farmers or small business owners (often both), usually located in rural villages, who own a mobile phone and rent use of it to members of the local community for a fee. For many people living in rural parts of Uganda, access to voice, text and data services can be difficult to attain due to the cost of mobile devices, cost of cellular service, availability of cell coverage, or difficulty using these services. Furthermore, since pay phones and Internet cafes do not exist in the rural regions of Uganda, the SPOs have stepped in to provide access to similar communication services, though they are more focused on mobile devices rather than desktops. Customers come to the SPOs when they want to make a call,

send a text message, or in some cases even surf the Internet and send email, and pay the SPO either per-minute or per-use. Monthly mobile access plans are uncommon in much of sub-Saharan Africa [2], so phone owners must purchase credits in advance that may be applied towards minutes and/or data use in a pay-as-you-go system. Since many of the SPO's clients are often not literate, the shared phone operators sometimes will type and read messages for their clients. SPOs are thus an important information gateway for rural customers.

A typical usage scenario of Grameen's system involves a farmer going to an SPO to seek information about a particular problem with one of his crops (e.g., spots on the leaves of banana plants). The SPO helps the farmer formulate a query to the agricultural tips database maintained by AppLab. An SMS message containing "banana leaf spot" is sent to a local number that provides access to the database search service. The service replies with a message containing what it determined to be the most relevant entry in the database given the query, and the SPO can then provide this information to the customer for a fee.

During the initial trial phase, Grameen made the SMS answering service free to the shared phone operators (though operators were free to charge customers), and asked in return that the shared phone operators administer a survey to their clients about their experience with the service. With the surveys AppLab collected feedback from the clients about the user experience, relevance and perceived usefulness of the information provided, what the users inferred about the provenance of the information, and how much they trusted it. In addition, they also wanted to get a sense of what users would be willing to pay for such services as they strove for a locally sustainable business model.

At the time of our deployment in November of 2008, no smartphone had been officially released in Uganda. In fact, Android phones had yet to be officially released in the United States. Most of the SPOs we worked with owned basic Nokia 1100 "candy-bar" style phones or equivalent models. A few used even more primitive


(a) Nokia 1100 (b) Nokia N72 (c) MTN Avvio

Figure 4.1: Phones used by SPOs before ODK

Avvio phones provided by MTN (a national mobile carrier in Uganda), and even fewer used higher-end Nokia N72s. All of these phones can be seen in Figure 4.1. The Avvio phones were limited in functionality to only making phone calls. The Nokia 1100 phones added basic texting capabilities, had a standard 12-key number pad, and used either multi-tap or T9 [59] systems for text entry. The Nokia N72s were more advanced and additionally provided a color screen, a camera, a web browser, and four-way arrow navigation buttons.

Before deploying ODK, AppLab's trial of their SMS system involved 117 SPOs spread throughout the country collecting the survey information on paper. Though they asked all of the SPOs to administer the surveys, AppLab would only collect the

paper surveys once a week from the 17 located physically closest to the central office. The process of sending a worker out to collect the paper forms, returning to the office, and entering it into a computer system took two days. Over the course of a week, AppLab would also call around 40 SPOs on a rotating basis to check on the SPOs and record any survey information. This also gave the SPOs a chance to give feedback about the services and ask questions.

### *4.1.2 Reasons Grameen Wanted to Upgrade from Paper Surveys*

For AppLab, it was important to deliver the surveys to the customers as they received the tip to get the most accurate responses. Paper surveys traditionally used for this purpose could not always provide the just-in-time surveying needed. SPOs often provided tips serendipitously while not at their office, and administering surveys for these tips would be infeasible as it would require always carrying the paper surveys.

Additionally, Grameen wanted to cut down the cost incurred from collecting the paper forms. The cost of collecting the data was primarily the combination of fuel and person-hours since collecting the forms required a worker driving for an entire day as they visited each of the 17 SPOs in the regions around Kampala. It took another worker 2-3 days to call another 40 SPOs to get survey information over the phone.

Finally, while not designed into their original efforts, the ability to capture agricultural problems with photos and GPS tagged locations prompted Grameen to begin pondering more advanced uses for their system, and to more easily detect and document problematic outbreaks.

Our goal working with Grameen in this deployment was to determine if ODK was suitable for data collection in developing regions. Since none of the SPOs had used a smartphone or a touchscreen before, and most had not even seen a QWERTY keyboard, we wanted to determine how much training was required for lightly educated users. Additionally, we wanted to identify potential problems deploying such a system, determine if an organization found added value from the system, and determine

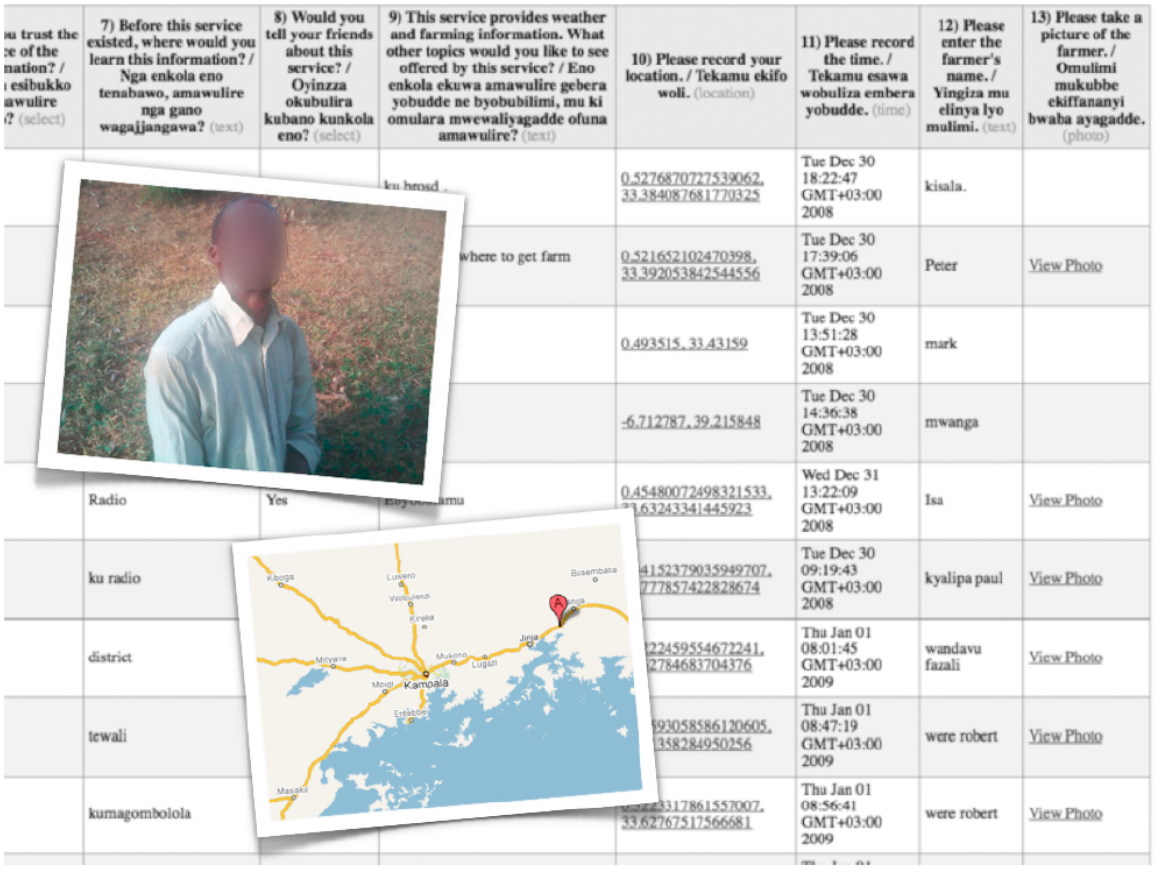
potential sustainability of using the software. Lastly, we sought feedback about the features an organization working in these regions would find desirable.

## *4.2 Implementation*

The initial version of Collect and Aggregate we used for this deployment significantly differed from the final versions discussed earlier in Chapter 2 of this dissertation. Before using XForms, we built the system using standard HTML web forms with special JavaScript "hooks" to access native functionality on the phones such as the camera and GPS. The web pages were downloaded from Aggregate to the phones and stored locally for offline access. Similarly, completed instances were saved locally, and a separate process was used to asynchronously upload the submissions when the phone detected network connectivity.

Prompts in the initial version were limited to text, select-one (radio buttons), photo, or location. Several fields not visible to the user containing device-specific information such as time stamps, SIM card identification numbers, and device serial numbers were also available as options. Forms could only be presented linearly and all questions were displayed vertically on a single page. The web-pages and text were fixed sizes and did not allow the user to zoom in or out. Though the system met Grameen's needs for this deployment, we found several limitations and received valuable feedback from Grameen about requested features that we discuss later in Section 4.3.6.

The initial version of Aggregate had a form designer built in, and created a database for each form at the time the form was created. It was not possible to build a form externally and import it. Aggregate displayed information in a table format with hyperlinks leading to other pages displaying binary data like photos. Data could only be exported in .csv format or streamed directly to Google Maps. Figure 4.2 shows an example of the features provided.



| ...u trust the ...e of the ...nation? / ...esibukko ...awulire ...? (select) | 7) Before this service existed, where would you learn this information? / Nga enkola eno tenabawo, amawulire nga gano wagajjangawa? (text) | 8) Would you tell your friends about this service? / Oyinzza okubulira kubano kunkola eno? (select) | 9) This service provides weather and farming information. What other topics would you like to see offered by this service? / Eno enkola ekuwa amawulire gebera yobudde ne byobubilimi, mu ki omulara mwewaliyagadde ofuna amawuliire? (text) | 10) Please record your location. / Tekamu ekifo woli. (location) | 11) Please record the time. / Tekamu esawa wobuliza embera yobudde. (time) | 12) Please enter the farmer's name. / Yingiza mu elinya lyo mulimi. (text) | 13) Please take a picture of the farmer. / Omulimi mukubbe ekiffananyi bwaba ayagadde. (photo) |
|---|---|---|---|---|---|---|---|
| | | | ku brosd ... | 0.5276870727539062, 33.384087681770325 | Tue Dec 30 18:22:47 GMT+03:00 2008 | kisala. | |
| | | | ...where to get farm | 0.521652102470398, 33.392053842544556 | Tue Dec 30 17:39:06 GMT+03:00 2008 | Peter | View Photo |
| | | | | 0.493515, 33.43159 | Tue Dec 30 13:51:28 GMT+03:00 2008 | mark | |
| | | | | -6.712787, 39.215848 | Tue Dec 30 14:36:38 GMT+03:00 2008 | mwanga | |
| | Radio | Yes | ...amu | 0.454800724983215 33, 33.63243341445923 | Wed Dec 31 13:22:09 GMT+03:00 2008 | Isa | View Photo |
| | ku radio | | | ...4152379035949707, ...777857422828674 | Tue Dec 30 09:19:43 GMT+03:00 2008 | kyalipa paul | View Photo |
| | district | | | ...22459554672241, ...2784683704376 | Thu Jan 01 08:01:45 GMT+03:00 2009 | wandavu fazali | View Photo |
| | tewali | | | ...93058586120605, ...358284950256 | Thu Jan 01 08:47:19 GMT+03:00 2009 | were robert | View Photo |
| | kumagombolola | | | ...2233178615570 07, 33.62767517566681 | Thu Jan 01 08:56:41 GMT+03:00 2009 | were robert | View Photo |

Figure 4.2: The initial version of ODK Aggregate that provided a tabular view of the information, access to images, and ability to create maps in Google Maps.

By leveraging Android's intent architecture, we were able to slightly modify Collect to open a survey at the time an SMS was received from Grameen's services, but not when other SMS messages were received. Upon receiving a message from one of Grameen's phone numbers, Collect would intercept the message, launch the survey, and populate the top of the survey with the message contents. That way, the survey was already open when the shared phone operators read the message, and they were less likely to forget to survey their customer. Received SMS messages were still saved in the phone's inbox for later review.

We worked with AppLab to deploy twenty HTC G1 devices to SPOs in different regions across Uganda. Within each region, Grameen chose SPOs based on the number of customers that typically utilized their SMS services. AppLab also balanced the number of SPOs whose storefronts had access to the power grid to those without access to get feedback about the battery-life of the phones. They had initial concerns that the smart phone devices would use more power than basic phones and not last sufficiently long for those working off the power grid.

The G1 phone, shown in Figure 4.3, provides a capacitive touchscreen, slide-out full QWERTY keyboard, microSD storage, 3.2 megapixel autofocus camera, accelerometer, digital compass, GPS receiver, and 3G and WiFi radios. An on-screen "soft" QWERTY keyboard appeared when the physical keyboard was obscured behind the screen. Internally the G1 has a 528 MHz processor, 192 MB of RAM, and a 1150mAh battery rated for 406 hours of standby or 5 hours and 20 minutes of talk time. The processor speed and memory of these devices is comparable to laptops commonly available in 2004, merely 5 years before this deployment.

At the time of our deployment, a representative of MTN mentioned that voice traffic was given priority over data traffic, so during peak usage data connections may be only intermittently available. Since our uploading process happened asynchronously in the background, the possibility existed that large amounts of data could be used attempting uploads that could be cut off and need to be retried later. To mitigate this


Figure 4.3: G1 phone used by SPOs testing ODK Collect.

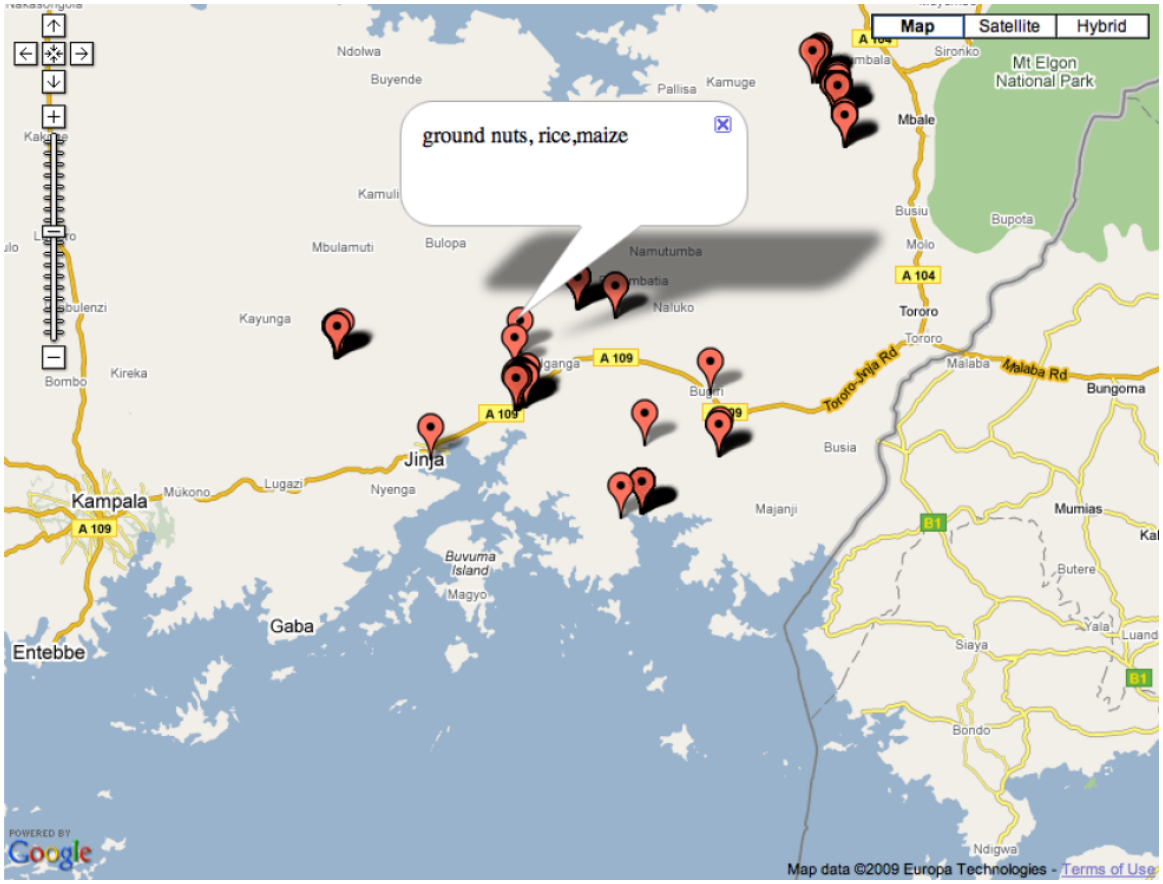



Figure 4.4: Map showing locations of SPOs working with Grameen in Uganda, built from survey responses about what crops farmers are growing.

problem Grameen provided the SPOs involved with enough mobile credits to qualify for unlimited data plans for the duration of the pilot.

The phones were deployed in phases, starting with ten SPOs in a single region near Jinja. After a training session covering two full days, SPOs were asked to use the system for approximately one week to test functionality. We visited each of the SPOs during this week to ask about any problems they had encountered. Once the initial usability issues were resolved, the remaining SPOs in other regions were trained in small groups a few weeks later. Figure 4.4 shows a map of the deployment locations in Eastern Uganda of SPOs from the first training session.

Figure 4.5: Training SPOs using a projection of the Android Emulator

### *4.2.1 Training*

For the first training session, 10 SPOs met with us, representatives of Grameen, and representatives of MTN in a conference room central to their businesses. Since none of the SPOs had used a smart phone before, and all but one had never used a QWERTY keyboard, we began by training the SPOs about the basic phone functionality such as making calls and sending text messages. Although the official language of Uganda is English, Luganda and similar variants are more common among the rural populations. Thus, training manuals for SPOs were produced in both English and Luganda. Similarly, all of the survey questions were presented on the phones in both English and Luganda, which fortunately uses the Latin alphabet. Given the number of survey responses received in Luganda, there is strong evidence that multi-language support is preferred. We also employed a Luganda/English translator for the training sessions. We found the most useful training method to bridge the cross-language gap was projecting an image of the Android SDK emulator (shown in Figure 4.5) where we could mime interactions with the phone. The entire training session lasted for two, eight hour days.

Following the basic phone usage training, we showed the SPOs how the surveys would automatically be displayed when an SMS message was received from the tip service, and instructed the SPOs on how to complete and submit surveys. To ensure full comprehension, SPOs role-played scenarios on the entire process of sending a text message, receiving a response, administering a survey, and submitting the response. A few days after the training a representative from AppLab followed up with the SPOs to ensure full comprehension and answer any questions that may have arisen.

### *4.2.2 Timeline*

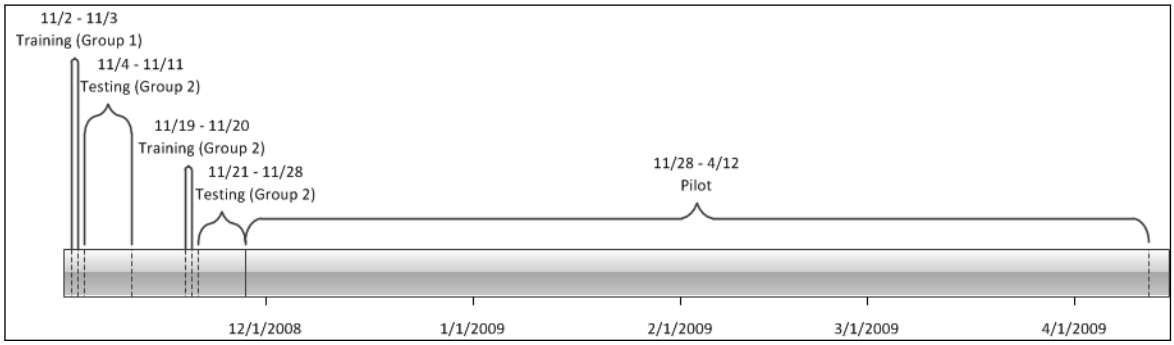



Figure 4.6: Timeline of the Grameen deployment

Figure 4.6 shows the entire timeline for our deployment. The training was split into two groups, each with two full days of training. Each time, the following week we instructed the SPOs to use the phones on their own. During this week a member of our team visited each SPO, giving them a chance to tell us about any problems and ask any questions. The pilot then ran for a little over three months.

## *4.3 Results*

Over the four months of the pilot deployment, Grameen collected 1106 surveys from the shared phone operators using Collect, and considered it a great improvement over their previous paper process. For AppLab, the large rural user base in Uganda is

an untapped wealth of information, and the ability to survey these users is critical. ODK allowed AppLab to, as noted by a project manager, "structure the app around the last mile before going to market."

During the deployment and training we recorded observations about the SPOs use of the phone. At the end of the deployment Grameen administered a paper survey to 13 of the SPOs for feedback on Collect, the G1 phones, and about their experiences using each of the phone and paper surveys. The survey contained open ended questions and asked the SPOs to state their preferences, if they had any, about using either system. We also interviewed the project manager for feedback about the entire system. All of the responses and feedback we gathered are summarized below.

### *4.3.1 User Feedback*

To gauge the usability of the phone and software, SPOs were asked if they had any problems calling, texting, or using Collect. Only three, all users in their mid to late 40s, reported minor trouble using the phones. Though our sample size was too small to draw definitive conclusions, we observed that older users were not as comfortable with the phones as younger users. However, all of the SPOs said they would recommend the device to friends and family. None of the SPOs reported issues with shortened battery life as compared to their previous phones.

### *4.3.2 Input Challenges*

The interaction with the touch screen was new to all of the SPOs. One asked, "Is there blood in the phone?" as he felt the phone must be alive since it responded to his touch. Many SPOs had significant difficulties with the touchscreen not responding to touch during the training sessions. After some experimentation, we determined that because several SPOs had calloused fingertips, their skin could not trigger the capacitive touchscreen. We determined that using the pads of their fingers solved the problem, but using their pads also introduced a more awkward touch interaction. To

help facilitate this new interaction, we redesigned the entire user interface with much bigger buttons to provide bigger target areas for finger presses.

Another issue we observed during the training was that several of the older SPOs held the phone at arm's length in an attempt to read the screen. Some of them had glasses, which helped alleviate the problem, but others said they could not afford glasses. To make Collect more accessible to those with poor vision, we significantly increased the font size, but not beyond a level that the other users found disturbed the general aesthetics.

After using the phone for several months, all but one SPO thought the touchscreen was easier than a standard 12-key keypad for entering phone numbers, and all of the SPOs preferred the QWERTY keypad to a standard 12-key keypad for sending text messages. One SPO felt the keyboard buttons were a bit close together while another said the touchscreen sometimes would not respond. So although the majority of the SPOs had never used a QWERTY keypad or a touchscreen and many had experienced problems with input during training, all had grown to prefer the touchscreen and QWERTY input.

### *4.3.3 Phone Versus Paper Surveys*

After having used both the phone and the paper surveys, every one of the SPOs reported a preference for the phone surveys to the paper surveys, with most citing ease of use. One respondent summed it up with, "It's easy, surveys are made and submitted instantly." Another compared it favorably to paper with, "It takes less time for a person to take the survey than paper writing." The efficiency of the process was also rated highly since the SPOs were no longer required to keep track of the paper forms. One responded said, "The G1 survey process is real-time as opposed to the paper forms where we had to wait for a month to be picked up by AppLab."

When asked how Collect compared with paper, the supervisor agreed with the SPOs. He preferred the real-time delivery of the surveys from SPOs as compared

to the slower collection and entry of the paper forms. One benefit not anticipated was the safety of data once collected. The supervisor preferred Aggregate because it "would guarantee [the paper surveys] not being taken to the shredder for anyone who was not aware of the data significance."

### *4.3.4 Improved Feedback*

The low latency of the survey responses proved very helpful in providing accurate and updated information that could be reviewed and acted upon quickly. The supervisor noted that, "action has taken place in improving the structuring of the tips given the recommendations from the SPOs who interact with the farmers on a day to day basis." He also noted that, "ODK has given AppLab a new outlook on how best to structure the apps to ensure that what comes to market is actually not off the mark with the users' expectations." With their old paper forms, it was not possible to update tips daily in response to the survey responses.

### *4.3.5 Safety*

The safety of SPOs carrying an expensive phone has been a concern voiced by NGOs with whom we had previously discussed deploying our system. When asked if they had any problems carrying and using the phone in public, SPOs reported no serious concerns. In fact, two SPOs reported enjoying the extra attention garnered from others' curiosity about the phone. One said, "The public always asks me on how I got the phone, how much it costs, also they want to learn more about its functions." In the case of SPOs, the more attention one can draw, the more customers one gains.

### *4.3.6 Design Limitations*

After examining the data collected over the course of the pilot, we determined sources of error and created a list of features that could help provide higher quality data.

**Branching:** Since the HTML forms could only be displayed linearly, the branching logic had to be built into the forms. For example, many of the questions within the form started with, "If yes, ...". These questions often made the form significantly longer than needed for a given set of answers during a form-filling session, and sometimes the lack of control resulted in fields being recorded that presented contradictory results, such as if the previous answer was, in fact, "no". Embedding question logic into the form and providing automatic navigation could prevent most of these errors.

**Multiple Languages:** Of the 10 SPOs in the initial training, only half could read and speak English well, so we needed to include both English and Luganda in the form. For this deployment, we simply included both languages in the question text. For example, "Have you used this service before? / Wali okozesezzako enkola eno?". Though this helped, sometimes it created long questions that required the user to scroll the question text off of the screen in order to access the answer field. This occasionally led to scrolling too far, all the way to the next question's answer field, where the user would erroneously enter the information for the previous question. Users would also occasionally lose their place in the form when the question text was not visible.

**Constraints:** Without answer constraints and more enforced data types, Grameen would occasionally find text answers in fields that should have been numbers, numbers answers which were out of range, and required questions left blank. By pre-specifying acceptable answer ranges and data types, many typographical errors can be caught at the time of data entry, making it easy to ask the user to correct them at that time.

**Navigation Issues:** When describing on-screen navigation we would often describe user interface widgets as "buttons". The SPOs initially found this very confusing since there was no physical button to press, leading us to instruct them to "touch the square on the screen with [specified words] in it". This lead us to explore using more natural gestures such as a swipe, which mimics turning a page in a book, rather than rely on icons for navigation.

**Embedded Multimedia:** Grameen felt it would help to embed multimedia into forms and answers. For example, if an SPO asked a question about "banana leaf spot", the service could send an image along with the response so that in the survey the farmer could see what banana leaf spot looked like and determine if that was the actual problem or if they needed to ask a differently phrased question.

With the exceptions of navigation and embedded multimedia, which we added later, we realized that all of these issues could be resolved by switching to an XForm based system. This feedback led us to make the important decision to implement the OpenRosa subset of XForms to provide a more powerful user experience.

### *4.4 Summary*

Our deployment with Grameen showed that ODK is a suitable platform for developing world data collection applications. The problems the SPOs encountered with the touch-screen user interface revealed that users in developing regions may need special considerations for accessible user interfaces, and it is important to test mobile systems in advance with each specific user base. Additionally, piloting these systems in the field is important since issues like calloused fingers are use cases that developers are unlikely to uncover working in a lab in a developed setting.

Though we received mostly positive feedback from Grameen, it was clear that our initial design was limited in its ability in regards to multiple languages, constraints, navigation, and multimedia. This feedback led us to stop using pure HTML forms and towards implementing the more powerful XForms standard to provide the desired richer interaction, greater flexibility, and higher quality data.

At the conclusion of the pilot project, AppLab requested an extension of the pilot to trial three additional surveys. When asked why, the supervisor noted that, "the enormous number of responses received during this pilot gave us much better insight on the SMS program." He felt if they could duplicate the response rate of the pilot for their other projects then it could provide more information about the impacts of

their other interventions. This ability to generate large amounts of timely feedback is an important aspect provided by ODK.

ODK provided Grameen with immediate information throughout the deployment process. First, we were able to analyze the entries from the training session to determine how well the SPOs understood the system. Next, throughout the deployment the immediate feedback gave Grameen a good sense for how often and how well the system was being used, allowing them to make continuous changes to improve their system, and determine sources of differing usage patterns.

After their successful piloting, the SMS services built by Grameen and Google were officially released in late 2009 under the name Google SMS [56]. They have since gone on to add other services such as an SMS based marketplace, providing a venue for customers to buy and sell goods. Additionally, they have begun conducting a social impact assessment to determine the effects of their system.

Perhaps the greatest indicator of success of the system emerged when we asked if AppLab would continue to use ODK after the pilot. The country director was unequivocal in his answer, "Put simply, the ability to quickly extract rich information (i.e. GPS and picture) quickly and with strong reporting provides reliable and immediate field feedback, which is critical to all of our efforts. This is preferable to both the SMS survey methods we use in other outreach and the paper methods we have used in the past, though I'd say ODK has probably put us off paper for good."

# Chapter 5

# DEPLOYMENT 2: EXAMINING COST, EXTENSIBILITY, AND INTEGRATION WITH EXISTING SYSTEMS

## *ACADEMIC MODEL FOR THE PREVENTION AND TREATMENT OF HIV, KENYA*

We partnered with the Academic model for Prevention and Treatment of HIV (AMPATH) in order to compare the cost of implementing a smartphone-based system using ODK to the cost of PDA/GPS and paper-based systems in the context of a community health worker program. Additionally, we demonstrated the extensibility of our system by adding a barcode scanner as a new feature to Collect, and we showed the benefits of ODK's modularity by integrating Collect directly with AMPATH's existing data storage system, OpenMRS. Finally, we surveyed users about their satisfaction using ODK compared to previous PDA and paper-based systems.

Our results from this deployment show that the cost of deploying ODK for an HIV testing, treatment, and counseling program attempting to reach two million patients in a rural catchment area was $0.13 per patient as compared to $0.15 per patient for a PDA/GPS system and $0.21 per patient for a paper-based system. By successfully adding a barcode scanner to Collect and integrating Collect directly with AMPATH's medical record system we managed to streamline AMPATH's data collection process, allowing them to provide improved services and focus on future expansion. In a post-study survey, we also found that AMPATH's users preferred ODK to their previous PDA and paper-based systems.

## *5.1 Background*

AMPATH [4] is the one of the largest HIV treatment programs in sub-Saharan Africa and is Kenya's most comprehensive initiative to combat the virus. The program aims to reach two million individuals in the program's catchment area, shown in Figure 5.1, with the following specific goals: (a) counsel and test all eligible individuals for HIV; (b) identify pregnant women not in ante-natal care; (c) identify orphaned and vulnerable children; (d) determine immunization status for children; (e) identify people at high risk for TB infection and collect sputum samples for testing; and (f) refer individuals for appropriate follow-ups when an issue is discovered. These goals are managed primarily through OpenMRS [88, 114], an open source patient medical record system often used by medical practices in resource-limited settings [84, 110]. AMPATH currently provides care to 130,000 HIV positive patients in the region at 26 urban and 26 rural clinics.

### *5.1.1 Current Processes*

In order to treat patients and populate their medical record system, AMPATH employs around 300 community health workers that it sends out into rural communities in pairs to do in-home visits with local residents. During these visits the health workers gather medical and family history, inform, test, and counsel about HIV, administer TB and Malaria tests (if the proper symptoms are present), screen for diabetes and hypertension, treat children for intestinal worms, distribute malaria-preventing bed nets, and provide nutritional counseling. At the conclusion of the visit the health worker records the patient's personal information on a health card and gives it to the patient. The health card has a printed barcode on the back so when the patient visits a clinic for medical care, the clinicians can scan the code and pull up the information on the patient gathered by the health worker.


Figure 5.1: A map of AMPATH's catchment area where it provides treatments with 26 parent and 26 satellite clinics.

Though all of the testing and counseling is voluntary for the patients, because of their extensive community involvement AMPATH community health workers have a 98% rate of acceptance into homes to do the counseling and testing.

Over the history of their program, AMPATH has evolved through several iterations of data collection systems. Originally, in 2001 they started providing comprehensive HIV testing and counseling with a strictly paper-based system. In 2008, they switched to an electronic data collection system using PDAs and GPS devices. For data collection they used Palm TX devices running Pendragon Forms software. GPS information was collected using eTrex devices, which were connected via cable


Figure 5.2: Palm TX, eTrex GPS, and connecting cable used by AMPATH prior to implementing a system with ODK.

to the Palm device. Figure 5.2 shows the PDA, GPS, and connecting cable used by AMPATH. By replacing their paper-based system with the PDA/GPS system they found that the per-patient cost of their program decreased from $0.21 per patient when using the paper system to $0.15 per patient with the PDA/GPS system [121].

### *5.1.2 Reasons AMPATH Wanted to Upgrade from a PDA System*

Though the cost savings of switching from paper to a digital system proved significant, several issues still lingered. First and foremost, AMPATH could not send the the data collected by the Pendragon system directly into their electronic medical record system. There were two reasons for this. First, the devices had no wireless capabilities and could not send data from the field. Thus, to retrieve the collected data the PDAs needed to be connected to a PC or laptop running the Pendragon software. This required workers to return to the clinic at the end of each day to offload the data. Second, the format Pendragon used to store data was incompatible with the Open-

MRS data model. Unlike many relational or tabular data storage systems, OpenMRS primarily stores values as coded concepts for easy search and analysis [122]. Hence, after copying the information from the device onto a computer, inserting the data into OpenMRS required several experienced data managers dedicating time to manually manipulate and insert the information.

The devices and software also had a few issues that proved problematic from a training and usability standpoint. The instruction needed to make users proficient in using the GPS units required a full day of training to make sure the health workers could retrieve an accurate GPS reading and transfer it to the PDA. Part of the reason for this was that the cable connection between the PDA and GPS devices was not always reliable and GPS information occasionally had to be manually doubly-entered into the PDA devices to ensure an accurate reading. AMPATH cited training as one of their most significant expenses in their program. During training AMPATH provides transportation and lodging for all of the community health workers. They also reported a fairly high turnover rate of community health workers. Thus minimizing training time could significantly reduce their project's costs.

Entering the identification numbers associated with the patient's barcode on their medical card was another common source of error. Without a barcode scanner, health workers had to manually enter the 10-digit numbers into their device. On several occasions AMPATH workers relayed a story to us about a six-year-old girl who presented at a clinic, but the computer brought up the profile of a 40-year-old man after scanning her card. It was later discovered that a single-digit typo caused this problem. Though this mistaken identity was easily detected, more subtle errors could result in potentially dangerous situations.

Lastly, the proprietary Pendragon forms software on the PDA devices limited flexibility to expand to new devices and incorporate functionality into the data collection software like barcode scanning.

AMPATH's goals in replacing their current system were as follows:

- Simplify the GPS process in order to both reduce errors, remove the double-entry requirement, and reduce training time and cost;
- Integrate the data collection process directly with OpenMRS to eliminate the extra step of having to manually convert the data;
- Determine if using a smartphone could further reduce the cost of data collection over a PDA system;
- Reduce data entry errors through constraints and additional features like bar-code scanning;
- Determine the usability of the device from the community health workers' stand-point;
- Produce a solution that can be customized and additionally modified to lower the cost of future implementations; and
- Produce a solution that works across a variety of handheld devices to facilitate the selection of hardware with the least cost for future implementations.

## *5.2 Implementation*

Replacing the GPS/PDA system involved several steps. First, we created an XForm that mimicked the functionality of the PDA system. The form AMPATH was using was primarily linear, but had a couple of simple branching steps that included, for example:

- Not asking males if they were pregnant;

- Only administering a TB test if the proper symptoms were present; and

- Only administering a third HIV test if the results of the first two were contradictory.

Since the Android phones we were using already contained built-in GPS units, this allowed AMPATH to replace both the PDA and GPS with a single smartphone device. Next, leveraging the Android "Intent" architecture we added a new barcode data type to Collect. Using the Intent system allowed us to leverage existing barcode scanning applications rather than forcing us to implement our own. For this deployment we chose the open source and freely available ZXing [127](pronounced "zebra crossing"), but any barcode scanning application implementing the proper Intent API would work as well. Figure 5.3 shows a community health worker using the barcode scanner to record a patient's card.

AMPATH decided to use the HTC G1 phone since it was the only phone available for bulk purchases at the time. The G1's specifications were discussed earlier in Section 4.2. The G1 can also be seen earlier in Figure 4.3.

Lastly, AMPATH created a module for OpenMRS that converted the XForm answers to the OpenMRS concept format. OpenMRS has a modular design that allows for adding external components that can communicate with OpenMRS' database to read, insert, or modify data. This change allowed AMPATH to use Collect's default mechanism of HTTP POST to send data, and only required adding the new module to the OpenMRS server to enable Collect to submit data directly into their OpenMRS database. In this way, they needed to only configure the phones with their OpenMRS server's IP address and the data was sent directly to OpenMRS, eliminating the need for the current intermediate data-entry step.

Figure 5.3: A HIV counselor from AMPATH in Kenya scans a patient's barcode with ODK Collect before sending data to the OpenMRS medical record system.

### *5.2.1 Training*

After implementing their Home-Based Counseling and Testing (HCT) form on Collect, we conducted two rounds of formal field-based usability testing. In the first round, we involved five pairs of two community counselor end-users. AMPATH felt five was an appropriate number since Nielsen has shown that approximately 80% of usability issues can be uncovered with approximately five subjects, with diminishing returns using more subjects [86]. One member of each pair was given a G1 device with Collect installed, while the other member used the existing PDA/GPS solution. These pairs were joined by two members of AMPATH's investigation team with clinical backgrounds and a member from the ODK development team. The pairs of end-users conducted HCT home visits for one day, while the investigators took notes on their actions and verbalizations. The health workers were asked to "think aloud" as they worked to assess efficiency and satisfaction with the new tool. We recorded any critical incidents, usability issues, software crashes, and difficulty meeting tasks.

At the end of each day's session, we conducted brief semi-structured interviews and a group discussion with the heath workers. These discussions focused on the usefulness and effectiveness of Collect as well as user satisfaction. The content of these interviews covered the strengths and weaknesses of the Android-based system when compared with the existing PDA/GPS solution. End-users were also invited to contribute corrections and comments on the interface design. Both investigators reviewed and agreed upon their assessments before their delivery to the application development team. Major usability issues uncovered during this assessment were addressed before final deployment of the application.

After we incorporated feedback from the first round into the application, we conducted a second round of usability testing and debriefing interviews in a similar fashion to the first round with the same number of providers. The second round of testing focused on making the interface more intelligent and user friendly.

### *5.2.2 Timeline*

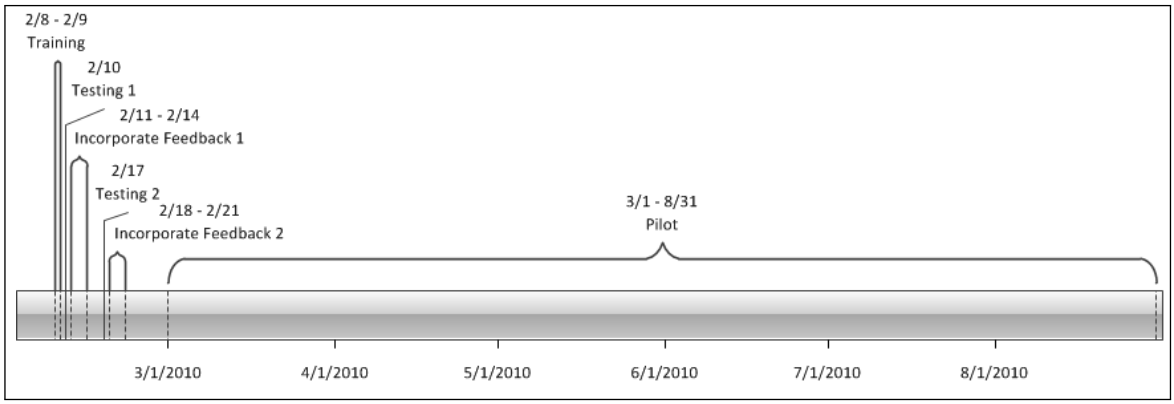



Figure 5.4: Timeline of the AMPATH deployment

Figure 5.4 shows a timeline of our entire deployment. We trained one group of community health workers over two days. After that training we had two separate one-day in-field testing sessions. The community health workers traveled to the field in groups of two where one member would use Collect and the other would use the old PDA/GPS system. They alternated who used which device at each visit throughout the day. We collected feedback during each of these testing days and spent a week after each incorporating the various feedback. The pilot then ran for six months.

## *5.3 Results*

Overall, AMPATH found ODK more cost effective than Pendragon Forms for both its software and hardware. During the first six months of using Collect they deployed Android phones with over 200 health workers and surveyed 18,108 households composed of 63,470 people using the initial implementation. They have since added their own user interface elements to speed up data entry and are moving to use ODK to help manage workflow using alerts and reminders [8]. They note that, "this is dependent on other infrastructure outside ODK...the possibilities are however limitless."

AMPATH found that, "the direct capture of electronic records greatly facilitated the expeditious performance of initial analyses and reports. Our work has highlighted ... most notably that only 28% of persons we are identifying as infected with HIV are presenting for follow-up care." AMPATH has acted on this data by launching programs to improve follow-up.

### *5.3.1 Cost*

After analyzing AMPATH's estimated cost of implementing the program, we determined that the cost of using the ODK-based system had been reduced to $0.13 per patient from the PDA/GPS cost of $0.15 per patient [102][1]. We note that the calculation assumes Android device costs available at the time of implementation, around $400 per phone. As of this writing, the most popular Android device in Kenya is the Huawei Ideos which retails for $80. Switching to this lower cost device would further reduce the cost to $0.08 per patient, nearly half of the previous PDA/GPS system.

### *5.3.2 User Survey*

To assess end-user attitudes towards the new Android-based data collection tool, we conducted an anonymous, self-administered survey of health workers who had used both the previous PDA/GPS solution as well as our ODK-based solution. Respondents were asked to evaluate the reliability of the solution as well as their satisfaction compared with the previously available tool. AMPATH administered the survey to the community health workers asking their opinions about the use of ODK as compared to the previous PDA/GPS system. At the time of the survey, 70 community health workers had used both the Android-based ODK Collect as well as the previous PDA/GPS solution. No members of the ODK development testing were present (or even in Kenya) during the survey. The survey was administered to a convenience

[1]The cost associated with the Collect implementation in the published report is listed as $.15, but there was an addition-error that was not caught until after final publication.

sample of 58 of these end-users during a weekly team meeting. Compared to the previous PDA/GPS solution and on a five-point Likert scale, end-users felt the Android system was faster (4.26), easier to use (4.43), and resulted in higher quality data (4.18). End-users felt using the smartphone system facilitated their interactions during home visits (3.98). Users felt that the training they received was adequate (4.21), and wished to continue using the Android-based system (4.47) compared with the earlier PDA/GPS system. Full results are shown in Table 5.1.

## *5.4 Summary*

AMPATH's users of ODK felt it was easier to use than their previous paper and PDA/GPS systems and facilitated their home visits. AMPATH also determined that ODK was more cost effective than PDA and pen-and-paper alternatives. Additionally, electronic data collection with ODK facilitated earlier reporting and allowed AMPATH to act on the data by launching programs to improve follow-up care. Once again, as with the Grameen deployment, immediate access to the collected information allowed AMPATH to continuously monitor and change aspects of their program, leading to improvements in their processes.

ODK's modularity allowed AMPATH to easily connect our mobile client, Collect, with their own patient record system running OpenMRS. This functionality was impossible with their previous system. Similarly, AMPATH was able to add a barcode scanner to Collect to further improve the data quality of the information collected by the community health workers. This allowed a practical demonstration of the flexibility of Collect's modular design over monolithic proprietary platforms.

Though the authors are no longer directly involved with the project, AMPATH has enjoyed continued success using our system and as of this writing several hundred community health workers have recorded over 650,000 patient visits using ODK.

Table 5.1: Responses to an 11-question survey posed to 57 community health workers of the HCT program in western Kenya

| **Question (Compared with using PDA/GPS system)** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|
| How reliable is the Android device in collecting data?$_a$ | 0 | 2 | 5 | 14 | 30 |
| How reliable is the Android device in collecting GPS coordinates?$_a$ | 0 | 2 | 3 | 8 | 37 |
| How do you rate the quality of data collected on the Android device?$_y$ | 0 | 0 | 12 | 17 | 21 |
| How easy is it to use the Android device to collect data?$_z$ | 0 | 2 | 4 | 15 | 30 |
| How fast can you collect data on the Android device?$_x$ | 1 | 1 | 5 | 20 | 23 |
| Does the Android device make it easier or harder to interact with individuals?$_z$ | 0 | 2 | 4 | 18 | 17 |
| Should end-users continue to use Android devices?$_{xx}$ | 1 | 1 | 2 | 16 | 31 |
| How easy is it to use the Android device to collect GPS coordinates?$_z$ | 1 | 1 | 2 | 9 | 38 |
| Was your training on using the Android device good or poor?$_{aa}$ | 1 | 2 | 2 | 24 | 19 |
| How much experience do you have using computers?$_{yy}$ | 10 | 0 | 30 | 3 | 7 |
| How much experience do you have with Android devices?$_{yy}$ | 40 | 0 | 6 | 1 | 2 |

$_a$ Scale: 1, very unreliable to 5, very reliable
$_y$ Scale: 1, much worse to 5, much better
$_z$ Scale: 1, much harder to 5, much easier
$_x$ Scale: 1, much slower to 5, much faster
$_{xx}$ Scale: 1, definitely discontinue to 5, definitely continue
$_{aa}$ Scale: 1, very poor to 5, very good
$_{yy}$ Scale: 1, never used to 5, regular expert user

# Chapter 6

# DEPLOYMENT 3: EVALUATING SUPERVISION, DATA QUALITY, USER INTERACTION, AND MULTIMEDIA

*ACTION RESEARCH AND TRAINING FOR HEALTH, INDIA*

The final deployment discussed in this dissertation was a three-way collaboration between PATH, an NGO based in Seattle, Action Research and Training for Health (ARTH), a non-profit organization providing medical care to the rural areas around Udaipur, India, and the ODK teamfrom the Computer Science department of University of Washington. Our main goal for this deployment was to run a study to thoroughly analyze the tradeoffs when implementing a data collection system with ODK as compared to ARTH's previous paper system. We sought to examine changes in time distribution at all stages of the process, accuracy and completeness of the data, impressions of the midwives and patients, and whether using videos in the forms as a teaching tool could improve patient comprehension and retention of important information.

In contrast to the previous deployments discussed in Chapters 4 and 5, we experienced several difficulties with this deployment that prevented us from achieving many of our previously stated goals. These difficulties included a key member of ARTH's team leaving for a new job, ARTH having difficulties updating some of their systems from outdated, incompatible software, and ARTH's inability to access the data collected with the new mobile system. Though unable to perform the rigorous scientific analysis we originally sought, we did learn several interesting lessons about both the use of the system and about potential issues that may arise within an organization when trying to move from a paper system to a mobile system. Recording user interac-

tions with the form provided important feedback about user comprehension, revealed insights into issues with the organization's processes, and enabled a new level of supervision. Additionally, we discovered that using videos embedded within the forms as an instructional tool provided the midwives with several reported benefits such as helping them better communicate across language differences, encouraging community involvement, and empowering them with more credibility when discussing proper methods for infant care.

Implementing a mobile system is not necessarily an exact substitute for paper and often requires changes to an organization's overall processes. While we discussed these changes before proceeding, we failed to ensure ARTH's comfort with these changes and that they had the appropriate resources to implement them. Demonstrating and testing each altered component before deploying the whole system may have helped prevent these issues.

Interestingly, even with the problems we encountered, ARTH is still pushing forward with integrating the mobile system. They see the advantages enabled by the system so far as outweighing the growing pains of switching from paper. We are continuing to work with ARTH to resolve the sticking points and develop local expertise so they can operate the system in a self-sustainable manner as our other partners have done.

### *6.1 Background*

ARTH [10] was established in 1997 to help improve the health of underprivileged communities around Udaipur. Their mission statement is, "to help communities access and manage health care according to their needs and capacity, by using research and training initiatives." ARTH operates three rural clinics, an urban health center, and primarily focuses on child health, reproductive health, and health systems and policy.

PATH [97] is an international nonprofit organization, founded in Seattle, WA in 1977, that seeks to transform global health through innovation. With active projects in over 70 countries their mission is, "to improve the health of people around the world by advancing technologies, strengthening systems, and encouraging healthy behaviors." Their goal in this project was to implement a mobile platform assisting midwives in developing regions, evaluate its use, and determine the requirements necessary to re-deploy such a system in other regions.

### *6.1.1 Current Processes*

Newborn and maternal health is critical in the days following a birth. Hence, ARTH operates a Post-Natal Care (PNC) program to ensure that both mothers and newborns living in the rural areas around Udaipur receive proper health care during this important time. In addition to providing care, the program seeks to educate mothers on proper continuing care techniques, reduce the rate of anemia, and provide ARTH a mechanism to gather information from their entire catchment area that they can analyze to reveal any concerning trends. The program is primarily conducted by seven midwives at three rural clinics. One of three doctors staffs the clinics one or two days a week to provide support to the midwives, but generally the midwives receive little direct supervision.

After a birth (at any health facility or home, not just in ARTH's clinics) is reported to ARTH, a midwife is dispatched within a few days to visit the mother and perform a PNC examination. If the birth occurs within one of ARTH's facilities, the PNC is conducted within hours, but more often the midwives must travel to remote villages to visit the mothers in their own homes. The PNC visit consists of the midwife taking an oral history, conducting an examination of the mother, and conducting a similar examination of the baby. Minor problems are treated on the spot, and if any major problems are discovered the nurses are instructed to refer the patient to a local hospital for immediate treatment. At the end of the visit, the midwife may provide

medication or iron tablets and are trained to discuss some of the techniques about proper breastfeeding, the importance of keeping the baby warm, and the importance of the mothers consuming enough iron either through diet or iron pills.

A second PNC visit, identical to the first, is conducted one to two weeks later. If no problems are found then no further visits are scheduled. In rare cases where problems still exist after the second visit, a third PNC visit may be scheduled.

Most of the nurses conducting the PNC visits have relatively little formal medical education, between one and two years of official training in school. As such, the PNC visit is conducted using a ten-page paper form, an example page can be seen in Figure 6.1. The form acts as both the data collection device as well as an instructional tool about what diagnoses to make given the data collected during the visit. For example, the form has spaces to record examinations like temperature and respiratory rate. At the end of the form there are fields with diagnostic logic stipulating, "Diagnose Complicated Fever if temperature is greater than 37.5C and any of the following is true...". The examination and diagnostic protocol is modeled after the World Health Organization's Guide to Child and Newborn Health [125]. It is important to note that given the structure of the form, midwives must either remember the information they recorded earlier or look back in the form to review the recorded information in order to make diagnoses. Often the logic for making a diagnosis is quite complicated and involves potential "and/or" combinations of many symptoms, making it easy to miss a combination for uncommon diagnoses.

All of the midwives carry a mobile phone with them to their visits, and are encouraged to call one of the doctors to confirm diagnoses and receive instructions on providing care when they find themselves unsure of how to proceed. Doctors and midwives alike reported that the midwives called on many visits where a potential problem outside of anemia was suspected.

The PNC form contains space to record information for up to three visits. Completed forms are stored at the rural clinics until they are collected, brought to ARTH's

| सं | प्रश्न | उत्तर | डिस्चार्ज कार्ड/हॉस्पिटल रिकार्ड को देखकर लिखें |
|---|---|---|---|
| 7 | क्या प्रसव के समय हॉस्पिटल रिकॉर्ड में हिमोग्लोबिन<br>• 6.9 ग्राम से कम<br>• 7 से 8.9 ग्राम के बीच में | हाँ/नहीं | 6.9 ग्राम से कम= Severe<br>7 से 8.9 ग्राम के बीच में= Less severe |
| 8 | क्या आपको गर्भावस्था के समय टीबी हुआ था ? | हाँ/नहीं | |
| 9 | क्या आपको गर्भावस्था के समय मलेरिया हुआ था ? | हाँ/नहीं | |
| 10 | क्या आपको गर्भावस्था के समय पीलिया हुआ था ? | हाँ/नहीं | |

| | क्या माँ को निम्नलिखित मे से कोई परेशानी है: | पहला दौरा | दूसरा दौरा | विवरण (कितने दिनों से, अन्य जानकारी) |
|---|---|---|---|---|
| 1. | आपको कैसा महसूस हो रहा है ? | | | |
| 2. | क्या आपको बुखार है ? | हाँ / नहीं | हाँ / नहीं | |
| 3. | क्या आपके हाथ पैरों में दर्द है ? | हाँ / नहीं | हाँ / नहीं | |
| 4. | क्या आपके बदबूदार स्त्राव हो रहा है ? | हाँ / नहीं | हाँ / नहीं | |
| 5. | क्या आपके पेशाब में जलन है ? | हाँ / नहीं | हाँ / नहीं | |
| 6. | क्या आपके पेट के निचले भाग में दर्द है ? | हाँ / नहीं | हाँ / नहीं | |
| 7. | क्या आपको बहुत अधिक कमजोरी है ? | हाँ / नहीं | हाँ / नहीं | |
| 8. | क्या आपके खून जाना जारी है? | हाँ / नहीं | हाँ / नहीं | |
| 9. | क्या माँ को दिन में 5 से अधिक पैड की आवश्यकता है ? | हाँ / नहीं | हाँ / नहीं | |
| 10. | क्या आपको प्रसव के समय बहुत अधिक रक्तस्त्राव हुआ था ? | हाँ / नहीं | हाँ / नहीं | |
| 11. | क्या आपको खाँसी है ?<br>• क्या आपको साँस लेने में तकलीफ है ? | हाँ / नहीं<br>हाँ / नहीं | हाँ / नहीं<br>हाँ / नहीं | |
| | • क्या आपके बलगम में खून जाता है ? | हाँ / नहीं | हाँ / नहीं | |
| | • क्या आप टी.बी के लिए दवाई ले रही हैं ? | हाँ / नहीं | हाँ / नहीं | |
| 12. | क्या आपके जननांगो में दर्द या सूजन है ? | हाँ / नहीं | हाँ / नहीं | |
| 13. | क्या आपके पेशाब का रिसाव है ? | हाँ / नहीं | हाँ / नहीं | |
| 14. | क्या आपके बच्चेदानी बाहर आती है ? | हाँ / नहीं | हाँ / नहीं | |
| 15. | क्या आपको उदासी महसूस हो रही है ? | हाँ / नहीं | हाँ / नहीं | |
| | • क्या आप किसी काम में ध्यान दे पा रही है ? | हाँ / नहीं | हाँ / नहीं | |
| | • क्या आपको जल्दी-जल्दी रोना आता है ? | हाँ / नहीं | हाँ / नहीं | |
| | • क्या आपको नींद आती है ? | हाँ / नहीं | हाँ / नहीं | |
| | • क्या आपको भूख लगती है ? | हाँ / नहीं | हाँ / नहीं | |
| | • क्या जो चीजें आपको पहले अच्छी लगती थी अब भी अच्छी लगती हैं ? | हाँ / नहीं | हाँ / नहीं | |
| 16. | क्या आपको रात में साफ दिखाई देता है ? | हाँ / नहीं | हाँ / नहीं | |
| 17. | कल से आज तक आपने क्या खाया ? | | | |

**B.** जाँच

| | | पहला दौरा | दूसरा दौरा | विवरण |
|---|---|---|---|---|
| 1. | तापमान | | | |
| 2. | नब्ज गति | | | |
| 3. | रक्तचाप | | | |
| 4. | साँस की दर | | | |



Figure 6.1: A page from ARTH's PNC data collection and diagnosis form.

main office, and entered into ARTH's database by a data entry clerk. The doctors do not review the information until it has been entered into their database. We found that, on average, the process takes 30 days from the initial PNC visit until the data is entered into the computer and ready for review. This lengthy delay from PNC visit to having accessible data prevents the doctors from being able to review the patient data collected by the midwives in a timely enough manner to schedule followups with patients presenting with any concerning symptoms.

For their database and reporting system ARTH uses EpiInfo 6, designed for the MS-DOS 6.22 operating system. However, Microsoft phased out MS-DOS with the introduction of Windows 95, over 15 years ago. The age of their database prevented importing of common data file structures like .xls or .csv, and all data had to be entered directly into the database manually by a data entry clerk.

### *6.1.2 Reasons ARTH Wanted to Upgrade from a Paper System*

At the onset of the project, ARTH's administrators highlighted several concerns in their current program that they hoped could be improved by switching to a phone-based system. The first issue they wanted to improve was to decrease the time from PNC visit to the time the doctors had access to the data. Their second concern involved reducing errors throughout the care process including both ensuring that the midwives were making proper diagnoses and referrals, and in reducing errors at the point of data entry into their database. Finally, they wanted to ensure that the midwives were counseling the mothers properly on techniques to keep themselves and their babies healthy.

With their current system taking around 30 days from point of collection to usable data in a database, ARTH's doctors were unable to review the data collected by the midwives in a timely enough fashion to schedule followups for patients presenting with concerning symptoms. Ideally, they could review the data every evening or every few days to ensure the midwives made appropriate diagnoses and referrals in order to

schedule any follow up visits if necessary. Currently, it takes potentially months to notice systemic mistakes in practice by a given midwife. With immediate feedback, errors can be much more quickly detected and corrected. Additionally, since the current system involves the data traveling between many people over several steps, errors can be easily introduced anywhere in the process. By correcting errors at point of collection and streamlining the process of data entry into the database they hope to reduce the occurrence of such errors.

Finally, two issues they had discovered in their patient population were high rates of anemia and that patients often performed traditional practices that were potentially harmful to newborns. To combat these issues, they instructed the midwives to counsel the mothers at the end of the visits about the importance of having a diet with enough iron and/or taking iron pills, the importance of exclusive breastfeeding and proper breastfeeding techniques, and the importance of keeping the baby warm through proper clothing and wrapping. Though ARTH had no hard evidence, they suspected that the messages being delivered by the midwives were inconsistent or absent in many cases.

In summary, ARTH's goals in switching to a mobile-phone based system were:

- Reduce time from data collection to data being available in the database so the doctors could provide supervision about the visits, make sure women were being referred properly, and provide feedback to the midwives.

- Improve visit counseling. ARTH hoped to standardize the information so all the women would receive the same message and hopefully be more likely follow the advice.

- Reduce data entry errors. The process of having a data entry clerk enter data from the hand-written forms was error-prone and time consuming. Data clerks would often have to call midwives asking for clarifications.

In addition, we sought to compare their paper-system to our mobile system on:

- The completeness (questions asked and examinations performed) during the visit.
- The effect of mobile device usage on time and activity distribution
- The capability of mobile device usage as an educational tool through videos, and determine retention rates as compared to current practices
- The attitudes and perceptions of midwives towards using mobile device as a mechanism to deliver routine care and treatment

### *6.1.3 Previous Attempts to Upgrade to Technology*

In our discussions with ARTH, we learned that they had previously attempted to partially replace their paper system by installing PCs in each of their three rural clinics. However, their attempt failed for several reasons. First, the remote clinics have very intermittent access to electricity, and even with 24-hour battery backups they found they could not keep the computers running consistently enough for the midwives to enter the data. Second, after a short time the midwives revolted against using the computers. They felt their duty was to provide health care and not to act as data-entry clerks. Since they still had to conduct the visits on paper and then enter the data into the computer, the system added to, rather than reduced, their total workload. This is an important distinction that we discuss further in Section 6.3.

ARTH felt that mobile devices could avoid the problems faced in their previous attempt because the phones did not rely on a constant power source, and that the data-entry could happen at the time of the visit rather than requiring a second data-entry step. Additionally, we hoped that the midwives would view the device as a tool that could help with visits and diagnoses rather than strictly as a data-entry step.

### *6.1.4 Baseline Study*

To establish a baseline from which to work, we gathered information about ARTH's current practices. First, we interviewed each of the midwives about their work experiences. Next, we sent an observer along with the midwives to observe 161 PNC visits and record the midwives' actions during the visits. Finally, for each mother visited, the supervisor re-visited the mother a week later to administer a survey relating to the care received as well as the advising provided by the midwife.

We interviewed each of the seven midwives about their current practices, common problems they experienced at work, and areas they felt could be improved. These interviews helped us establish our understanding about the roles and responsibilities of the midwives and about their perceptions of their role in patient care. The interviews took place in ARTH's head office and were administered by a member of PATH's India office.

After the interviews, we conducted an observation phase where a supervisor accompanied the midwives on their PNC visits. The midwives were not told the reason the observer was there except to, "observe the PNC process". The observer recorded information on a form nearly identical to the PNC form, however, instead of recording the mother's information, they recorded whether or not the questions on the form were being asked and whether or not the physical examinations were being performed by the midwife. The observer also recorded the start and end times for each section within the form. Additionally, the observers noted what advice, if any, was given at the end of the visit regarding taking iron, breastfeeding, and keeping the baby warm. This information gave us insight into both how thorough the midwives were during their visits using paper, and provided us with a time distribution of each section within a visit in addition to overall visit lengths.

As a last step, the supervisor revisited the mother five to seven days later and, with her permission, asked her questions about the PNC visit both relating to her

opinion of the care and of the advice given at the end. Our goal for this step was to capture the patient's perception of the care provided and attempt to determine the effectiveness of the counseling provided by the midwives.

### *6.1.5 Baseline Results*

The baseline results showed that most of ARTH's previously stated concerns were valid. We note that the data indicated that all of the visits in the baseline occurred at the mothers' homes. We expected a mix of home and clinic visits and believe there was a miscommunication about the intent of the study that caused the observers to focus only on in-home visits. Though we cannot accurately compare home to clinic visits, this actually simplified our analysis to the more common case. Additionally, none of the midwives' performances presented as statistical outliers so we present the data as an aggregate of all midwives.

PNC visits averaged 25 minutes each. We expected the second PNC visits would be somewhat shorter due to the fact that much of the demographic information had already been collected during the first visit, but second visits only averaged a minute shorter at 24 minutes each.

On average, four days elapsed between delivery and the first PNC visit. The second PNC visit was conducted on average 5 days later, and, on average, another 20 days elapsed before the data entry was completed. Thus, from birth to data in the computer took an average of 29 days.

In terms of thoroughness of the midwives during the PNC interviews, the observers noted that the midwives asked only 58% of mother interview questions, performed 71% of all mother examinations, and only performed 54% of child examinations. We expected that the percentage of questions asked and examinations performed would be potentially lower for mothers giving birth within ARTH's clinics because the midwife may have already known the information, but as noted previously, all of the visits happened at the mothers' homes. Conversely, we expected that the

Table 6.1: Comparing mothers who did and did not receive counseling information about the given topics with whether they reported having received it or not. n = 161

| | **Received Information** | **Remembered Info** | **Did Not Receive Information** | **Remembered Info** |
|---|---|---|---|---|
| **Iron** | 134 | 148 | 27 | 24 |
| **Breastfeeding** | 119 | 116 | 42 | 34 |
| **Baby Warm** | 64 | 93 | 97 | 31 |

presence of the observer would cause the numbers to be slightly inflated due to the observer effect [65, 75]. Thus, we believe these percentages are an upper-bound on completeness.

Relative to counseling, the observers noted discussions about iron in 83% of visits, counseling about breastfeeding in 74%, and keeping the baby warm in only 40%. Additionally, when the supervisor visited the women for a followup visit, the mother's reported remembering getting information about iron 92% of the time, getting information about breastfeeding 72% of the time, and getting information about keeping the baby warm 58% of the time. Interestingly, 88% of women who did not receive counseling information about iron reported receiving that information, 82% who did not get information about breastfeeding receiving getting it, and 32% of women who did not get information about keeping the baby warm reported receiving it. The actual numbers for these figure are shown in Table 6.1. The second set of results where women reported receiving the information when they did not suggests that the numbers of those who actually received the information and remembered is artificially high, as it is likely the mothers were telling the supervisors what they thought they wanted to hear [37]. Because of the rather large discrepancies in the data, we deemed the mothers an unreliable source for the purposes of our study.

We similarly discovered that ARTH's concerns about errors were also sound. In our baseline data we found 17 cases where the woman should have been referred to a hospital for immediate treatment, and 7 different cases where the baby should have been referred. Of those, no woman was referred. In fact, not a single referral for mothers or newborns was recorded for any of the 161 visits during the baseline.

While we had no direct way of detecting errors in the data entry process, we observed that several columns between the PNC form and our observer form contained the same information such as dates, locations of the visits, times, and ages. After eliminating any columns containing free-text like names where often differences were due to translation interpretations, we compared the remaining columns that should have contained exact matches and found that 12% of these entries contained differing values. Closer analysis of these errors suggested that most were typographic errors introduced at the time the data was entered into the computer.

### *6.2 Implementation*

Our first step in converting the PNC program to a phone-based system was to re-create the PNC form in the XForm format. Since the PNC forms were in Hindi, we hired a translator to translate the form to English. From there we created all of the prompts in English first, taking advantage of the OpenRosa itext structure to add the Hindi translation at a later time. We instrumented the PNC form using branching logic to automatically follow the work flow depending on the answers given. For example, one of the first questions asked if the mother had received an ante natal care (ANC) visit. If she answered "yes", several more questions appeared about the date and nature of the ANC visit. Otherwise, the form would continue to the next section. Similarly, we instrumented the form to automatically suggest diagnoses to the midwives based on the information they had previously entered. Those diagnoses requiring a referral were headed with a bolded "Recommend Referral" to encourage the midwife to refer the patient. We note that the paper forms had no information

suggesting which diagnoses required referrals and which did not. For all diagnoses the midwife was presented with a list of appropriate options to record what treatments they administered.

Though Collect provides the functionality to make every question require an answer, we opted to only make questions required that determined branching and to which the midwives could always have an answer such as, "Could you complete the PNC visit". Answering "no" to this question immediately took the midwife to the end of the form, while answering "yes" would guide them through the whole of the PNC interview. We refrained from making every question required to determine how often or if midwives would skip questions. The baseline data suggested that the midwives would skip questions and examinations on the paper form quite frequently, and we wanted to determine if that changed when using the mobile device.

In order to improve data accuracy, we did impose constraints on certain fields. That is, temperatures, pulse rates, ages, etc... had to fall within physically possible ranges. Values entered outside of those ranges would show a dialog to the midwife containing the acceptable range and prevent them from moving forward until the value was corrected.

### *6.2.1 Phone*

ARTH chose to use the Huawei IDEOS phones in this deployment. The IDEOS was the first low-cost smartphone released in developing countries, first in Kenya and then later in India. The phone contained all of the standard smartphone features: touch screen, wifi, 3G, camera, accelerometers, GPS, etc..., but used a smaller screen than most phones measuring in at only 2.8 inches. Though available in India at the time, ARTH could not locate any for sale locally and so we sourced them in the United States. We deployed a single phone at each of the three clinics since only one midwife from each clinic would conduct PNC visits on any given day. None of the midwives had ever used a smart phone before.

### *6.2.2 Modifications*

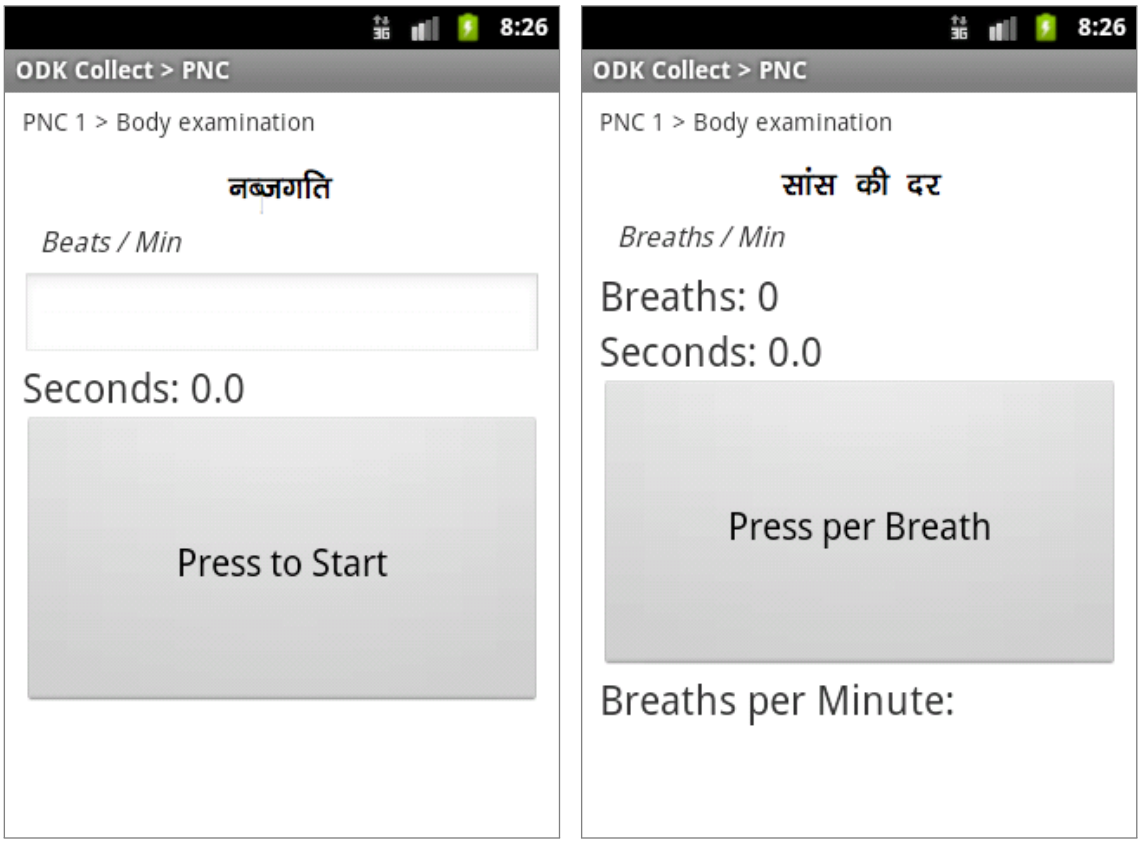
We modified Collect in several ways to more closely mimic ARTH's current processes and to collect user interaction data. Since they were only using one form, we modified the menu buttons to only have options for "PNC 1" and "PNC 2". Once a midwife completed the first PNC, the record would be stored using the mother's name and her husband's name (in English). Choosing PNC 2 would bring up a list of names of mothers that had completed PNC 1. Once a midwife completed PNC 2 the record would disappear from the menu but remain on the phone.

#### *6.2.2.1 New Widgets*

As part of their examinations, midwives needed to record information on pulse and respiratory rates. Normally, they wear or carry a watch to help with this task, but several times in the baseline data this information was not recorded due to the midwife forgetting to bring a watch with her on the visit. Hence, using Collect's modular widget design, we created two new widgets for this deployment to help the midwives with each of these tasks.

**Pulse Counter:** The first widget we added was a stopwatch widget for counting the patient's pulse, shown in Figure 6.2a. The widget provides two buttons: the first to start and stop the timer, and a second to reset the timer. When the timer reaches 60 seconds, the phone plays an audible signal and vibrates briefly to alert the midwife to stop counting. In this way the midwife can keep her focus on counting the patient's pulse rather than have to divert her attention to check the time. After counting, the midwife must manually enter the value.

**Breath Counter:** The second widget was a breath-counter widget, shown in figure 6.2b. This widget was created to help the midwives measure the respiratory rate of the mothers and babies. We modeled the widget after a similar device that had been previously proven successful by PATH. The widget consisted of a single button




(a) Pulse Counter (b) Breath Counter

Figure 6.2: Screenshots of new widgets created for ARTH

that the midwife pressed for each breath. The first breath started a timer that counted to 30 seconds. Each press of the button would increment a count displayed on the screen. Also, each press of the button caused the phone to vibrate briefly, letting the midwife know she had successfully hit the button without having to take her eyes off the patient. After 30 seconds, the phone produced an audible beep and a longer vibration to let the midwife know the counter was finished. At this point the count was automatically doubled to produce a breaths-per-minute count, and the button became inactive to avoid altering the count. The button became active again if the midwife cleared the answer. It is important to note with this widget that the midwife cannot manually enter a count.

#### *6.2.2.2 User Interaction Logging*

We added an additional set of logging to ODK Collect to capture the user interaction with the form. The original intent of capturing this information was to be able to compare timing information with that recorded in the baseline, however the interaction data turned out to be useful for much more as discussed in Section 6.3.1. For each interaction we collected the phone's timestamp and the action performed. For movements between questions the question and direction of movement within the form were also recorded. Broader phone-level events such as booting, shutdown, and changing of times and timezones were also logged. A complete list of all events logged is presented in Appendix A.

#### *6.2.2.3 Synchronization and Data Storage*

Of the three rural clinics operated by ARTH, two of them have overlapping catchment areas. Thus, ARTH requested to synchronize completed PNC 1 forms across phones so that the nurses would not be required to locate the same phone used in PNC 1 to complete the second PNC visit. We decided that in adding the sync we would use the same mechanism as the data store, replacing Aggregate in this deploy-

ment. We developed a stand-alone application on the phone to sync the forms using Dropbox [43]. Dropbox is a free cloud-hosted data storage service providing secure storage, transmission, and sharing of files on the web. Dropbox provides a robust API allowing users to build custom applications using their cloud service.

To synchronize, the application performed the following steps:

1. Download a list of all the files stored in two folders, "in progress" and "done".
2. Compare the list to the locally available files
3. Upload any local files not in Dropbox
4. Download any "in progress" files located on the Dropbox server but not present locally
5. Move any local files located in the Dropbox "done" list to a local "done" folder.

We noted in our development that collisions of forms modified on different phones could be possible. That is, two midwives update the same form on different phones before syncing. Dropbox does not automatically handle collisions, but does save revision history. Therefore, our order of operations ensured that all data was uploaded and all revisions stored. The conflicting files would be copied to a "conflict" directory, and we could use the revision history to manually resolve the conflicts. So far, no conflicts have been recorded during our deployment.

The synchronization application simply consisted of a single button. Upon pressing the button, the application would display a dialog updated with the current status of the synchronization, and report a success or failure when finished. The midwives were instructed to make sure the phone had a data connection, and press the button to sync twice a day, first thing in the morning and last thing at the end of the day. In this way, we could ensure that all phones would be updated with all forms at the start of each day.

Since the version of EpiInfo ARTH was using did not support importing external data in formats other than .rec files, they requested that we push the data into a MicrosoftSQL Sever database that they already operated. To make this possible, we developed a desktop application in Java using the same Dropbox API to pull the data from the cloud. The desktop application, however, could not upload or modify data in the Dropbox storage folders; it could only pull data into local copies. The data was downloaded in individual files in their original .xml format. One of ARTH's technical staff then wrote a short script that parsed the XML and inserted it into their database.

### *6.2.3 Images as Text*

At the time of our deployment no version of the Android operating system fully supported fonts for the Hindi language. Since most of the midwives did not feel comfortable interacting with the form completely in English, we leveraged Collect's the ability to embed images into the form. Text was translated from English into Hindi in a Microsoft Word document, then screen captures were taken of the Hindi script and saved to .png files. The PNC form was then created without text, but with images embedded in the question header. The results can be seen in Figure 6.3. The PNC form had over 350 different images representing script in Hindi. This method worked well for our deployment because we only used one type of phone. However, since other models have different screen resolutions the images on those phones could potentially be displayed much smaller or larger, and would require capturing the images at different resolutions for each device.

Android 4.0, released in January of 2012, fully supports Hindi. However, as it was unavailable at the time, our deployment would not have been possible without the ability to embed images. Additionally, many languages are not supported by Android that Collect can support through this technique of using images of text.

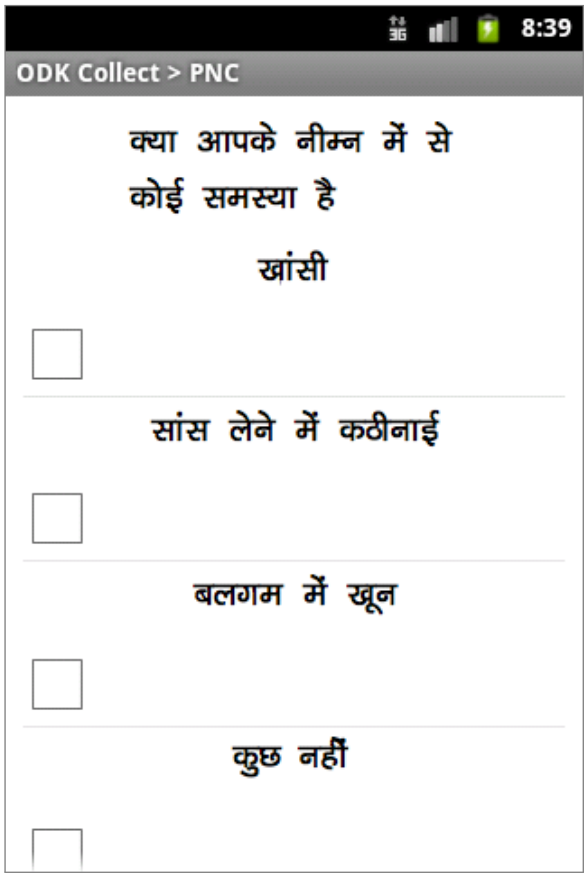



Figure 6.3: PNC form in Collect using images for Hindi script.

### *6.2.4 Videos*

In an attempt to provide a more consistent and memorable message, ARTH and Path wanted to create three short videos about the importance of ingesting enough iron, the importance of exclusive breastfeeding and proper technique, and the importance of wrapping the baby appropriately to keep it warm. We chose videos on the mobile devices because they have been shown to be useful motivational tools with community health workers in rural India by Ramachandran et al. from Berkeley [103] as part of the First Days project. The goal of First Days is "to create real impact in maternal and neonatal health in rural India by improving the efficacy of village-level health workers" [48]. In this project, they gave health workers mobile devices with two types of videos: 1) persuasive videos for pregnant women, and 2) testimonial videos targeted at motivating the health workers. They found that the videos were able to increase health worker self-efficacy and also provide modest learning gains. They found that creating videos with local context was important.

Rather than provide motivation to the health workers or the mothers, we sought to create educational videos targeted at the mothers and determine their effectiveness as instructional tools. The videos were all recorded on the same day, and each involved an ARTH midwife interacting with a local mother who consented to being in the videos. This conversational approach mimicked the approach pioneered by Digital Green for teaching agricultural information [54]. The video participants spoke in the local dialect of Mewari, and all the content was created by ARTH to be relevant to their PNC program. We tried to use as simple a setup as possible to film the videos so that the process could be easily recreated in the future by ARTH, Path, or other organizations. Our only equipment consisted of a videocamera mounted on a tripod and a corded stereo microphone attached to the camera and to the midwife's clothing. The videocamera was a Kodak Z18, and we used no additional special lighting or other equipment. Our setup is shown in Figure 6.4.

Figure 6.4: Our setup for creating the local videos with messages about taking iron, proper breastfeeding, and keeping the baby warm.

We used Windows Live Movie Maker (included for free with the Microsoft Windows operating system) to edit the videos. The final durations of the videos were: iron (2 minutes 28 seconds), breastfeeding (1 minute 42 seconds), and wrapping (2 minutes 24 seconds), for a total of 6 minutes and 34 seconds. The videos were inserted into the form to be played during the visit. The iron and breastfeeding video were to be played after the mother's diagnosis section. The wrapping the baby video was to be played after the baby's diagnosis section. Midwives were instructed to show the videos during every initial PNC visit and present it as an option during the second visits.

### *6.2.5 Training*

Each of the midwives received one full day of formal training on the phones, several weeks of phones being available for testing in the clinics, several weeks of conducting visits with both paper and the phones, and a final informal in-field training session.

For the initial training we split the midwives into two groups of three and four. During two consecutive days, one group came to ARTH's main office for training while the other worked at the clinics. The second day the two groups switched locations. All of the training was conducted in English with one of the doctors translating to Hindi. Since none of the midwives had ever used a smartphone before, especially one with a touch-screen, we began by training the midwives about the basic phone functionality such as making calls and sending text messages. Once the midwives were able to call and text each other, we instructed midwives how to complete the PNC form on the phone by stepping through the entire form and demonstrating the capabilities such as branching and diagnoses. Finally we taught the midwives how to sync the data using the Dropbox application. The midwives spent the remainder of the day conducting mock interviews on each other to further familiarize themselves with the phones and to give them an opportunity to ask questions if any problems arose.

The phones were then placed in the clinics and over the next several weeks the midwives were given the option to practice or use them for visits. This extra time was optional because we needed to update the forms and fix several bugs. During this time we received zero practice submissions, indicating that the midwives were not taking the extra time to practice with the phones.

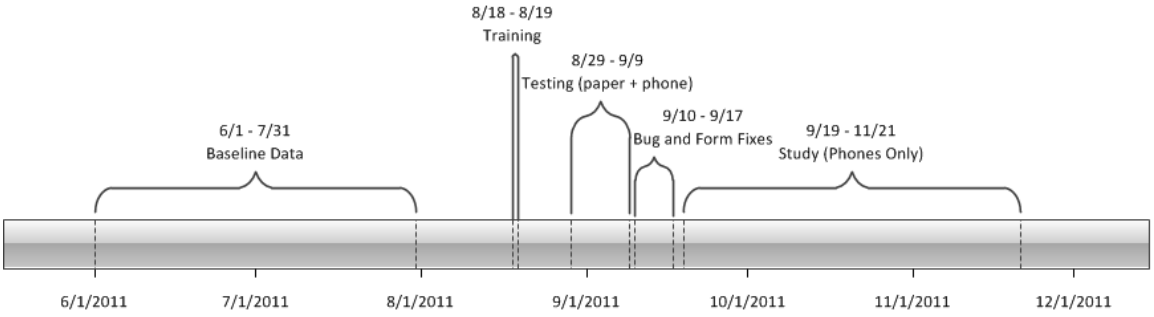
Once the phones had been updated, the midwives were instructed to take the phones along with them on their visits and enter the data on both the paper and in the phones. ARTH's administrators were worried about possibly losing data, so they wanted to have a several week period of collecting the data using both methods. Additionally, they hoped this would give the midwives more time to familiarize themselves with the use of the software.

Finally, the midwives were re-visited in the field one last time and given a final day of training. After the final day of training, the midwives were instructed to stop using the paper and to start using only the phones.

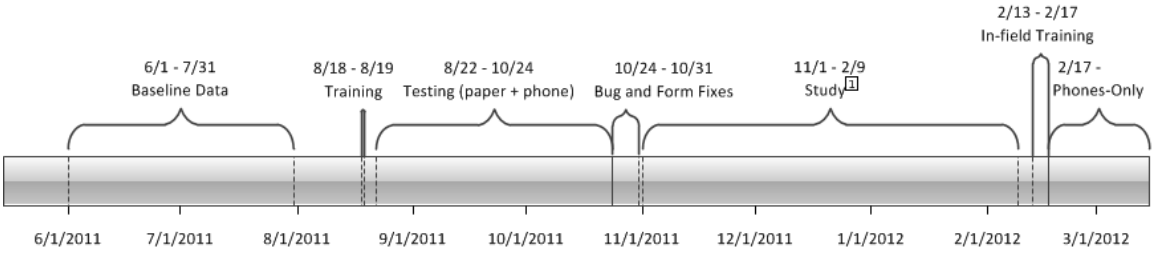
### *6.2.6 Timeline*

Figure 6.5 presents both the planned and actual timelines from our deployment. Our deployment with ARTH is still ongoing, but has strayed from its original timeline due to several complications. First and most unfortunate for the project was that the doctor assigned as ARTH's lead in the project (also the main midwife supervisor) quit just after the formal training sessions because she found a job located closer to her husband's work. ARTH has yet to replace her, and thus the midwives did not receive adequate supervision and support in their use of the phones following the initial formal training.

We discovered later that there had been a miscommunication between ARTH's administrators and the midwives about the purpose of the phones, and the midwives were unaware that the purpose of the phones was to replace the paper during the visits, causing several problems. First, only when ARTH instructed the midwives to stop




(a) Planned Timeline




(b) Actual Timeline

Figure 6.5: Timelines from our deployment with ARTH representing our planned and actual timeline.

using paper and switch to the phones did they bring up questions about differences in the workflow. Originally we were told, and observed, that data went onto the PNC forms and from those forms into the database, never being used for other purposes. As it turned out, the midwives took data from the forms to complete ledgers at each of the rural clinics and generated IDs from those ledgers that went back into the forms outside of the visits. This discovery further delayed the deployment while we worked with ARTH to resolve the issue. Second, without understanding the true purpose of the phones, the midwives viewed them as data entry device rather than a visit tool. This gave the impression of adding more work to their jobs without added benefit,

[1]We believed the phones-only portion of the study to be running at this point, but using the UI log data, discovered that the midwives were still using paper in addition to the phones

and thusly slowed adoption. Interestingly, this pattern followed almost exactly the same pattern as ARTH's previous attempt to install computers in the clinics.

Finally, ARTH has had trouble accessing the data collected with the new mobile system. Since their current system in EpiInfo6 did not support importing external data, and they have successfully switched several of their projects to MicrosoftSQL, ARTH asked us to send data from the phones into MicrosoftSQL. However, after the system was running we were told that none of the members of the PNC project had used MicrosoftSQL or knew how to how to analyze the data within it. In this case, we made an incorrect assumption that upgrading would not be problematic, and may have been able to correct this in advance of deployment by more slowly upgrading and testing each component individually. The inability to access the data so far has prevented ARTH from realizing many of the benefits from the system. As a compromise, ARTH has agreed to upgrade to EpiInfo 3.53, which supports importing both .xml and .csv files, and we are working to provide the data regularly in .csv format.

## *6.3 Results*

Due to the issues discussed earlier in Section 6.2.6, to date ARTH has yet to re-create the conditions of the baseline study with the phones, that is, with an observer present and followup interviews with the mothers. Hence, we do not possess the proper data to do a scientifically rigorous quantitative analysis. However, from our user interaction logging we do have analysis of the midwives' use of the phones. Furthermore, we have qualitative results from a survey administered to the midwives after they started using the phones exclusively during PNC visits. As this deployment is ongoing, our results focus on what we learned from the interaction data and what the system has enabled for ARTH.

### *6.3.1 User Interaction Data*

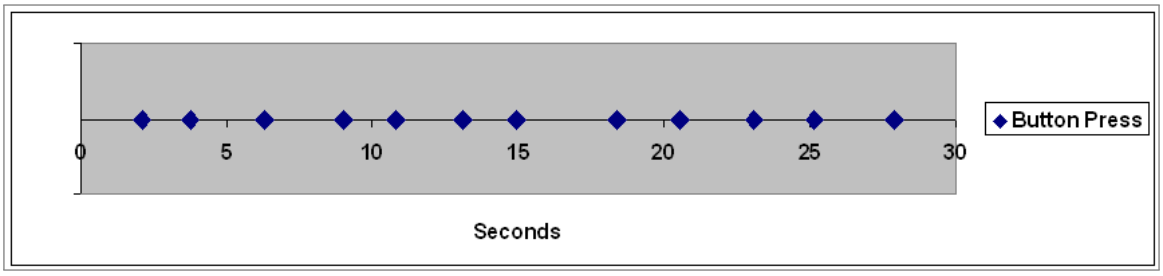
Though the original intent of capturing user interactions was to give us the ability to compare timing information about activity distributions, we discovered that the interaction data provided a wealth of information about everything from improper use of the forms to errors in ARTH's diagnosis protocols.

Once ARTH reported that the midwives had started using only the phones during their visit, we began analyzing the form and log data to make sure everything was operating smoothly. However, we noted several oddities in the data. First, we noticed that the average visit time dropped from 25 minutes using paper forms to only 20 minutes with the phones. This result contradicted our few in-visit observations where visit lengths with the phone had increased to around 35 minutes. Also, given that the inclusion of videos added nearly six and a half minutes, this mean that actual question and examination times had been theoretically reduced to a mere 14 minutes. We expected that by adding the videos, the visit length would actually increase. In another strange twist, we recorded cases where the visit length jumped to nearly two hours. Closer analysis of interaction during these long visits showed the midwife completing half a visit, putting the phone down for two hours, and returning and completing it later. The last concerning trend we noticed relative to visit timing was that often several consecutive visits started within one minute of the previous visit. This would be possible if the visit had occurred within a health center where many patients are located in the same building, however most of these entries were recorded as in-home visits. When we inquired about these oddities the midwives revealed that they were actually still doing the PNC visits on paper forms and entering the data into the phones either at the end of the visit or at a later time.

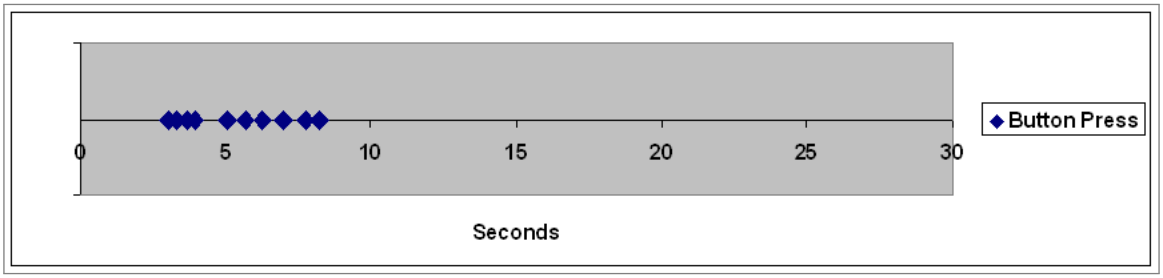
As we analyzed the user interaction data recorded during the form filling sessions we also noticed a couple instances of strange behavior. Periodically, as the midwife was nearing the end of the form and would reach a diagnosis, she would suddenly

navigate backwards through the form until she reached a question, change that answer, and then proceed back the diagnosis, which often did not display again as the value changed by the midwife usually changed the reported diagnosis. In our followup interviews we discovered two reasons for this behavior. The first was simply that upon seeing the suggested diagnosis, they realized they had entered an incorrect value earlier in the form and were returning to correct it. The second was that they disagreed with the protocol's diagnosis and changed the answer so the diagnosis would not appear. In an example of this behavior a midwife noted "painful urination" as reported by the mother, which combined with a fever lead to a diagnosis of a urinary tract infection. The midwife, however, determined the mother's pain was actually caused by stitches she had received after giving birth. This changing of the answer to match the diagnosis could potentially be dangerous because the data does not reflect symptoms reported by the patient, and rendering a doctor's review useless. At a higher level, repeated instances of users engaging in the same behavior may indicate an error in ARTH's protocol. These insights into the protocol and subsequent diagnosis would not be possible with paper forms and without monitoring the specific user interactions.

Another area where we detected strange behavior was with the breath-counter and stopwatch widgets. As these were the only two widgets that had interaction other than selecting or entering a value, their use proved very interesting. An expected usage of the breath-counter is shown in Figure 6.6a where the user has a fairly steady button-press rate over the 30 second she as they counts the patient's respiratory rate. However, an overwhelming majority of the time we would see something more like Figure 6.6b where many presses happened in the first 5-10 seconds and the midwife would simply wait for timer to expire. During follow-up interviews the midwives reported several interesting reasons for this behavior. A couple of the more confident users said they were simply multitasking. They would record the breaths while conducting a series of examinations, and simply wanted to enter the number rather than take another count. This multitasking suggests that less structured in-




(a) Expected Usage of Breath Counter




(b) Unexpected Usage of Breath Counter

terfaces [63, 38], where the midwives choose the order of examinations and questions may be preferable and better enable multitasking during the visits. Others reported being uncomfortable with the widget because it was hard to hit the button while watching the patient breathe. Several also reported they felt the 30 second interval was too short to ensure an accurate count. The stopwatch widget provided useful information in a completely different manner, in that not a single midwife ever used it. We suspected this was a training issue, and follow-up interviews verified our hypothesis. All of the midwives reported not knowing or remembering how to use the stopwatch widget, so they chose to use other methods. Interestingly, the midwives did not think to mention these problems to us until we specifically asked about them, and without the data showing inconsistencies we would not have known to ask.

Though we could not pinpoint the exact causes, having the user interaction data available allowed us to see anomalies in the data, giving us cause to ask the midwives about specific behavior. The answers revealed a variety of causes from not understanding how to use the software to attempts to quicken the visit process. There

exists an increasing body of work analyzing anomaly detection in data sets, however most of this work focuses on only the collected data, specifically form data in our case [22, 119, 66, 96, 101]. Applying these techniques to data about the users' interaction at the time of collection can potentially provide additional context, revealing errors that form data alone cannot [14].

### *6.3.2 Patient Reaction*

Similar to the midwives not receiving enough explanation about the phones' intended use, the patients often did not receive a thorough explanation from the midwives when using the phones. In the follow-up interviews, midwives reported patients frequently believing the midwives were "using Facebook" or "playing games" on the phone rather than focusing on the patient. It was not until the nurses gave the patients the phone to watch the videos that they understood they were using the phones as tools. This misunderstanding could have been initially prevented by instructing the midwives to explain the phones, or by adding a reminder prompt within the form. However, the midwives reportedly began introducing the phone after experiencing that reaction from patients.

### *6.3.3 Time*

The ability to receive timely data from the PNC visits was very important to ARTH, and with our system they were able to lower the time it took for data to be available from an average of 30 days down to data being available the same evening as the PNC visit. As they integrate this new data availability into their workflow, doctors will be able to more closely monitor the midwives' performances and schedule followup visits when necessary.

Once they stopped using the paper forms and switched to only using the phones, the midwives felt that they were spending more time in visits, but that they were also spending much less time doing paperwork at the clinic. They felt the decrease

in paperwork more than made up for the increase in visit times, and that it allowed them to provide better care. The midwives also mentioned they felt the longer time in the visits was beneficial due to the fact that phones caused them to be more thorough with the patient because the phone presented the questions one at a time.

### *6.3.4 Increased Confidence*

After using only the phones to conduct the visits, the midwives indicated that they felt improved confidence to act. By presenting potential diagnoses, the phone validated the midwives' analysis and presented them with a list of proper actions for each diagnosis. This also caused them to make fewer phone calls to the doctors seeking information and validation. The doctors similarly noted receiving fewer calls once the phones were in use. Doctors also received fewer calls due to the diagnoses presenting specific treatment options for each case. The paper forms did not present the treatment options specific to each diagnosis, so midwives often felt the need to verify their chosen treatments.

### *6.3.5 Videos*

Of all the practices that changed after we deployed the phone system, the midwives were most excited about the presence of the videos. While primarily meant to convey information for the mothers' benefit, midwives found that the videos helped bridge language barriers, encouraged community involvement, and gave the midwives more credibility with the mothers and other relatives of the patient.

Midwives felt the videos conveyed the information much better by letting the patients see in addition to hear. One noted, "By talking I can't let them understand everything, but by video she is showing what is necessary, taking iron and how to feed the kids." Another commented, "The patients feel the video is really good because they are watching it practically. Before we were telling them how to do things and now they are seeing it, so they are learning." Similarly, midwives felt the level of

detail being presented was important, “It is very detailed in the video. Before we let them understand things, but it was not like this, we could not let them know everything, it was not as detailed.”

Much like the results reported by First Days, we found local context to be extremely important. On midwife commented, “In the video when we show the patient, the patient understands that there is one more patient like me in the video and one is the nurse and she is letting her know how to take care, so she feels that ok she is also a patient like me and how I need to take care of myself and my baby. So if there will be two nurses sitting in the video, then she will not understand those things as much.” Previously, class differences between the midwives and patients often prevented adoption of information. By seeing a patient exactly like themselves in the video, they were more likely to relate to and trust the information.

In an unexpected result, midwives felt the videos gave them more credibility and built trust with the patients. One of the midwives mentioned about another younger midwife that, “If there is a nurse, like [name], who is very young, the mothers don’t take her seriously for telling them you need to take care like this. They think that she is younger, she doesn’t know, and she is not married. But there are so many younger nurses, by showing the video, there is an elder woman in the video, and so they think ok, so there is some doctor telling us this.” Similarly, another mentioned that, “By showing the video, the patient thinks it’s necessary to take the iron pills. They understand what I am saying, so they trust me more than before”. Again, social factors like age and class were mitigated by the presence of the videos.

The dialect spoken around Udaipur is primarily Mewari, and nurses from other areas mentioned having trouble conveying messages to the mothers. However, playing the videos helped them bridge the gap. One reported, “There is some language problem between me and the community so the patients can understand by looking at the video what I want to tell them.” Another noted, “I enjoy showing the video. When I talk to the villagers they don’t understand, but by showing the video they

understand. I am from...and my language is different from here. At the beginning I was taking the driver to help me tell the patients." A third mentioned, "I am having language problems because I am not from here, not local, so the video helps me a lot by telling the patient what to do. Because the local people sometimes don't understand my language."

Finally, one midwife noted that there was more community interest and involvement on her visits when she showed the videos, "When we were doing PNC before, only the patient and I were available there and I was saying everything to the patient and now I am showing the video, the other people listen and they come also and like the mother-in-law and other family and they also learn how to take care of the patient and they are also learning." Figure 6.6 shows an example of a midwife playing the video for a mother with an interested crowd onlooking.

Though the responses were mostly positive, there were a few areas where the midwives felt the video detracted from their interaction. One noted, "Now when I am showing the video, I can't talk in the middle of the video, it is a communication gap." Another mentioned that, "The communication is less than before because of the video. All the things they are watching in the video, we are not interacting." Though the videos could be paused, exploring a more interactive technique with more midwife involvement could be useful in the future.

### *6.4 Summary*

Though ARTH encountered some growing pains in attempting to upgrade from paper to a mobile system, they feel the benefits demonstrated so far by our mobile system outweigh the trouble. Most important to them is the decrease in time to available data, from 30 days down to one. This immediate feedback will give them the ability to better supervise the midwives and ensure that patients are receiving the best care possible. We are continuing to work with them to make sure they can successfully operate the system and that it will be sustainable for them over the long run.

Figure 6.6: Curious community members join in watching the educational videos.

Our deployment with ARTH demonstrated the benefits of collecting user interaction data to provide important feedback about user comprehension as well as insights into issues within an organization's processes. The data on how the user interacts with the form provides a new, comprehensive tool to supervise workers and was previously unobtainable with paper-based and current mobile systems.

Additionally, we found that presenting patients with educational videos facilitated interaction by helping midwives bridge language gaps, established trust with the patient, and encouraged community involvement. Some midwives did report that they felt they had less communication when using the videos, showing the need for future exploration in this area. More interactive video techniques involving embedding questions or having the midwives pause more often to interact as in a facilitated learning technique could prove useful. Also, the mostly favorable feedback about the video interaction coincides with previous work using videos for education, but exact impact is still difficult to measure [6], and more work is needed to determine if they result in an improvement in the quality of care.

Though we were unsuccessful in in completing many of our measurement goals, we learned several interesting lessons about both the use of the system and about potential issues that may arise within an organization when trying to move from a paper system to a mobile system including loss of adequate supervision during training periods, not properly educating users on the purpose of the system and eventual deployment, and issues integrating into a given workflow. Mobile systems are not an in-place replacement for paper, but they must be integrated into an organization's overall processes, requiring changes to other components. Understanding the magnitude and pre-testing the changes is important to ensure a smooth deployment.

# Chapter 7

# CONCLUSION

Organizations working to improve the lives of the over one billion people living in poverty throughout the world rely on remotely collected data for everything from decision making to impact assessment. Thus, the ability of these organizations to collect data in a timely manner, assure the accuracy of the data, analyze the data to find meaning, and present summary data to spread awareness defines, in large part, the chances of success for any given campaign. While organizations in developed regions have realized improvements in their data collection processes through technology, organizations in developing regions have been unable to achieve the same gains due to insufficient infrastructure, hindered by lack of access to wired communication and electricity. The sharp rise in access to mobile devices that operate unhindered by the current infrastructure limitations provides a potentially rewarding path forward for mobile data collection systems utilizing advanced smartphone devices that deserves further exploration.

In this dissertation we have examined the components of data collection campaigns, investigated the challenges faced by organizations attempting to conduct these campaigns in developing regions, and presented the added value provided by implementing these campaigns with mobile systems built on smart phones. Specifically, we presented and discussed our contribution to the space of data collection systems, Open Data Kit. We showed through analysis of the design, along with evaluations of several real-world deployments, the benefits created over existing systems by the newly available capabilities realized in ODK.

Our first contribution centers around the ODK system artifact, presented in Chapter 2. To our knowledge ODK is currently being used by organizations in 29 countries on 6 continents, and Collect has been installed on over 10,000 devices worldwide. Since the project is completely free, open source, modular, and decentralized, these numbers are a lower bound as we have no way to effectively track every installation. Our numbers have been gathered from Google Market's statistics, downloads from our own web page, and organizations that have contacted us. The Open Data Kit community now consists of over 500 individual members and over 200 developers who have exchanged over 5,000 messages in the three years of ODK's existence.

For our second contribution we showed through examples how ODK's extendible, modular system design has given organizations the ability to easily modify and customize applications, utilize external components, and integrate with existing systems. Chapter 3 discussed a variety of modifications the community has contributed, extending and adding new components to the system. In a more detailed analysis, Chapter 5 discussed how AMPATH specifically took advantage of the extensibility by adding a barcode scanner and leveraged the modularity by enabling the mobile devices to submit directly to their own data storage system, OpenMRS. Finally, in Chapter 6 we discussed our own experiences adding new user interface widgets to help midwives count respiratory and pulse rates.

Our third contribution focused on demonstrating the effects of ODK on the key criteria specified by organizations working in developing regions: cost, time, supervision, training, and data quality. We evaluated these criteria through each of our real-world deployments. First, Chapter 4 laid out the criteria we used to design a system for better data quality, and Chapter 5 and Chapter 6 discussed how the implementation of those decisions improved the data collected by making the process easier and removing possible sources of error in the workflow. Additionally, in Chapter 5 AMPATH showed a decrease in overall cost using ODK over PDA and paper systems for a community health worker program. Training time was also significantly reduced

with the health workers only having to learn to operate a single device with an easy to use GPS interface. Finally, Chapter 6 discussed ARTH's significant improvement in latency from point of data capture until that data was available in a database, improving from an average of 30 days down to one. Furthermore, ARTH was better able to remotely supervise their midwives thanks to the ability to collect user interaction data in addition to the improved immediacy of access to the data collected in the field.

Finally, Chapter 6 discussed our last contribution, presenting promising additional benefits made possible by ODK, including new directions to explore using multimedia as first class objects in forms and the ability to record user interactions and timings with forms. By embedding videos into the form itself ARTH was able to standardize messages on the importance of breastfeeding, iron intake, and proper wrapping of newborns delivered to the community by the midwives. The midwives felt the videos helped them better communicate across language barriers, catalyzed conversations about the issues, encouraged greater community involvement, and gave them more credibility with the mothers. Also, by recording user interactions with the form we were able to capture unexpected user behavior, possibly indicating potential problems. Further investigations of these behaviors revealed a variety of causes such as lack of user comprehension, errors in the protocols themselves, and workers not properly following expected workflows. This previously unattainable information can give organizations new insights into their own practices and the effectiveness of their workforce.

## *7.1 Retrospective*

In examining the history of the development of Open Data Kit, we made many decisions along the way that significantly impacted the directions we took the system. This section examines the decisions we felt had the most impact, and we hope will provide insight to those building the next generation of data collection systems.

### *7.1.1 Android and Smart Phones*

When we first started developing Open Data Kit, many working in the data collection space expressed trepidation when we mentioned we were targeting Android smart phone devices. Cost and availability were cited as the primary reasons since the original phones retailed for nearly $500 and most organization's target price was closer to $100 per phone. Many also predicted that it would take several years for smart phones to penetrate into rural markets. In late 2010, just over a year after we started developing ODK, the first Android smart phone was officially released in Kenya at a price point of just $80. Additionally, as we began testing the system, organizations realized they could increase savings in other areas like training and transportation by leveraging the advanced features and sensing capabilities of these more powerful platforms. As newer devices such as tablets running the same Android operating system have come to market, organizations using ODK have been able to upgrade seamlessly since the application requires no modifications to run across different devices running Android. Though Android smart phones have yet to see wide adoption by the general public in many developing regions, organizations seem to be moving away from using PDAs and J2ME feature phones for their workforces.

With the explosion of available Android devices, building a native application has proven to be a wise choice. However, this decision also prevented us from easily expanding to other platforms such as Blackberry and Apple iPhones and iPads. Though members of the community have begun building compatible applications for these platforms, building a more web-based system based on HTML5 and Javascript could bridge the cross-platform issues. In this way, the developer needs to only develop a wrapper around a web-view for the platform specific interactions. We had originally considered a similar approach, but at the time the local-storage and execution components of HTML5 had yet to be fully implemented and impeded our goal of always-available offline access.

### *7.1.2 Cloud Technologies*

Our original intent in creating a cloud-based data storage system was to enable organizations to easily test ODK, and to be able to create data collection systems without needing the IT overhead of owning and maintaining local web-connected servers. The idea of having a system running for dollars-per-month rather than the startup cost of nearly $2,000 for a server in addition to ongoing costs of power, security, and maintenance seemed ideal for many organizations. However, we found several major setbacks, including organizational politics and the immaturity of cloud-technology, that prevented widespread adoption and caused us to develop a system that could be run locally as well.

Organizations, especially those working with sensitive information like personal patient data, prefer to be the keepers of their own data. Many feel with cloud technologies that they lose ownership of their data because they do not know exactly where their data resides. Using a service like Google App Engine means the data gets replicated many times within a data center and possibly across multiple data centers located in different countries [23]. As some countries have laws preventing patient information from leaving the country's borders, attempting to use such a system is technically illegal. Similarly, though the replication provides excellent service and backup guarantees, little precedent has been set about the ownership of the data. For example, the possibility does exist that a government could seize all of the data in one of these centers and claim ownership since it resides within its country's borders.

As a counter example, we heard a tale from an organization working in one of these regions that originally kept all of its records on one computer with no backups. One day a service man arrived at the organization and informed the person working at the front desk that he had been called to perform some maintenance on the computer. After a brief examination, the service man decided he needed to take the computer back to his office to fix it. He left the office with their machine in hand, no questions

asked, never to be seen again. In this case, it is unlikely the thief was after the data as much as the machine's hardware, but the example does illustrate some of the unforeseen dangers of storing all of the information locally. Viruses, backups, and availability are other issues that need to be considered.

Another issue we encountered related to the maturity of the cloud technologies. Though our experience was primarily with Google App Engine, these lessons can be applied across other cloud implementations as well. First, we found imposed system limitations to be a problem. Applications developed by third parties for deployment in these cloud systems run on what is essentially rented equipment. As such, the hosting organizations have gone to great lengths to protect their infrastructure, other programs operating within it, and to ensure availability. To make sure they do not exceed their own capacity they impose limitations on access, throughput, and storage, however, these limits are ever increasing. For example, initially App Engine only supported file sizes of 1 megabyte, much smaller than the size of images produced by the phones' cameras. Eventually, however, the limit was raised first to 10 megabytes and later to 30MB. Second, with App Engine under constant development internally, we were continuously forced to make modifications to coincide with changes in processing, limits in storage, and even API changes and cost changes. Keeping up to date with these constant changes required diligent developer maintenance and constantly pushing new releases to users, hindering our goal of not requiring a developer to maintain the system.

There are many tradeoffs in terms of cost, maintenance, and security when using cloud technologies versus locally maintained machines. Both have their rightful places depending on the needs, politics, and resources of each organization. Even with cloud technology's growing pains, we believe it is likely still the best place to provide many of these services for a majority of organizations going forward. However, with the intense restrictions and safeguards for private personal data like patient information, local systems will still be necessary for the foreseeable future.

### *7.1.3 OpenRosa*

Another controversial decision, using XForms and the OpenRosa standards, proved to be one of our greatest assets before eventually becoming our biggest hindrance. Initially the OpenRosa compatibility provided our system with good momentum within the NGO community as many organizations were already using or familiar with OpenRosa compatible systems. Additionally, we were able to establish good relationships with other development organizations, especially Dimagi, and learn from others who had been working in the space. However, eventually the combination of OpenRosa's implementation and the structure of XForms left us with limited options going forward.

The original OpenRosa library was implemented specifically for J2ME phones, and to maintain backwards compatibility has limited capabilities in terms of storage and memory options. Currently, entire forms must be processed in order to build large tree-structured data models that must reside in memory. Loading the entire data model into memory limits the sizes of the forms we can process, even with the more powerful smartphone architectures. We were able to increase the speed of some of the loading of forms by caching serialized versions of the forms locally, but the memory problems persisted.

Similarly, though the declarative nature of XForms makes them more comprehensible, we have found the structure provides limited options for further customizations. Extending the OpenRosa code to include new widgets and/or data types requires modifying the underlying library for each new addition. This requires the community to agree on the new additions, or forking to a customized version, potentially fragmenting the ecosystem. Similarly, OpenRosa does not currently expose all attributes associated with all tags, limiting the scope of possible extensions.

While it is impossible to determine whether or not we would have received as warm a reception from the community had we initially set out with another standard, either

our own or a different open standard, given the opportunity to begin again we would design for more flexibility and customization than OpenRosa provides. We believe that providing an easy to use, yet powerful form designer, combined with the ability to integrate submissions with other systems, renders the choice of underlying form language moot.

## *7.2 Future Work*

Though Open Data Kit has provided much real world value to many organizations, we believe it is just the first step in data collection systems built using advanced mobile devices. As the price of advanced technology drops and the scale of deployments increases, new directions will need to be explored to further improve the data collection process. Based on our work, our own observations during deployments, and feedback given by partner organizations, we present several promising future directions in the space.

### *7.2.1 Integrating the Capture of User Interaction*

Our experiences have shown us that the ability to record users' interactions while entering data can provide a great benefit to an organization monitoring their data collection processes. While we recorded the interaction data separately for exploration at a later date, integrating the reporting of the user interaction data directly with the associated form data could provide a much richer data set for organizations to analyze and understand. The availability of the user-interaction data at the same time as the form data could help reveal issues about misunderstanding of forms, reveal gaps in the process, or detect malicious falsification of data [14].

### *7.2.2 Advanced User Interface Customization*

The most common request we received about ODK was about customizing the look and feel of different prompts within a form. While we were able to add many of these

using the "appearance" attribute in XForms, we found many requests not generalizable enough to fit into the ODK framework. The "body" tag in XForms does include a description of what the form should look like, but it is presented in a markup language and leaves the exact display and layout open to the renderer's interpretation, in our case, ODK Collect. Thus, currently the only way to modify the look and feel of each widget involves editing the source code to either change or add a new widget.

In future systems, a better design would be to provide default widgets, but also allow users to specify inputs, outputs, and data types as well as a way to specify a file describing the exact look and feel for the widget. In this way, users can customize fonts, sizes, colors, and any other attributes they deem necessary. Likely this system would be a hybrid between the current ODK and the original ODK we presented in Chapter 4 using a web-page-like structure to define the interface with "hooks" into the device's functionality using a combination of HTML, Javascript, and CSS.

### *7.2.3 Sensor Integration*

Most smart phones today come equipped with many integrated onboard sensors such as GPS, cameras, magnetometers, accelerometers, and microphones, vastly increasing the information that can be collected with these devices. However, new devices are being pushed with even more sensing capabilities. For example, the Galaxy Nexus phone comes with near field communication (NFC) technology, and many newer model phones support connecting USB peripherals such as pulse-oximeters and ultrasound [20, 24] devices, converting phones into portable medical stations. Supporting USB peripherals allows phones to connect to a nearly limitless combination of sensors, but future systems need to be designed to take advantage of these capabilities.

Already, many sensing systems integrated into the environment around us can capture information as broad as weather [64] or as local as a person's movements [81]. By enabling communication with these systems, many new capabilities can be added without additional cost or adding complexity to the phones themselves.

### *7.2.4 Data from External Sources*

We built ODK primarily as a system used to fill out blank forms and submit the completed data to a centralized server. However, many organizations wanted the ability to pre-populate fields, especially when collecting several forms about the same subject such as a patient. CommCare [30], whose mobile client for Android is built using ODK, implements some of these features, but their approach is very specific to health care and not readily generalizable. ODK Tables [68], a spreadsheet client designed to lower barriers experienced by information providers in the developing world, provides the ability to pre-populate data into a form used by Collect at the start of a data entry session, but we have yet to enable populating fields by pulling from external data sources when completing a form.

### *7.2.5 Dynamic Itemsets*

Though ODK supports branching based on answers to questions, often organizations desire to go a step further where the contents of a prompt change based on a previous answer. An common example of this feature would be to present a list of all 50 states. Upon selecting a state, Collect would present a list of counties within the state. Upon selecting a county, Collect would present a list of cities within the county and so forth. This static example is fairly common, however, cases exist where the contents of each prompt change over time and require pulling the corresponding data from an external source.

### *7.2.6 Bulk Phone Setup and Management*

As the scale of data campaigns increases, managing phones will become a more prevalent issue. ODK puts the onus on users to download and/or update forms and send data back to the server. However, organizations often would rather push form updates out to the phones and have data automatically transmitted back to the data storage

system. Allowing the phones to register themselves with the data storage system and then allowing administrators to control updates to groups of phones would eliminate many update issues. We have taken the preliminary steps towards designing and piloting a system called ODK Manage to try to solve these issues.

### *7.2.7 Advanced Multimedia*

ODK supports both embedding and capturing multimedia such as audio, video, and images as first-class objects. Though this has proven effective in deployments covered in this dissertation, we feel there is much more to be explored related to the advantages of multimedia. For example, embedding questions within a video, stopping the video until the user answers correctly, or determining which video to show next based on given answers could provide a richer user interaction that also ensures the user understands the target concept before continuing. Segmentation of the videos may also help alleviate the concerns related to a lack of person-to-person communication noted by ARTH's midwives in Chapter 6. Additionally, partitioning could better enable the teaching interactions envisioned by Johns Hopkins as discussed in Chapter 2. Finally, at a larger scale, the effectiveness of each portion of a video could be analyzed by the frequency distributions of related answers.

### *7.2.8 Tasking*

Though ODK can provide real-time data to administrators from the field, it provides no mechanism for easily tasking or re-tasking workers in the field. Luckily, the device we targeted is a mobile telephone and re-tasking can often be administered with a single call. However, a truly integrated system could support task lists, calendaring, and follow-up visits, if necessary. These features could make organizations more agile, minimize overhead, and reduce duplication of effort. Leveraging its close integration with Collect, ODK Tables is able to provide some of these features.

### *7.2.9 Completing the Feedback Loop*

Perhaps the area that future systems could improve on most relates to completing the feedback loop. Admittedly, ODK is designed as a one-way data collection platform. That is, workers gather data in the field and send it to administrators for analysis. The feedback we received most often from field workers was that they rarely received feedback about their own work. Often times workers collecting information saw neither the higher level view nor the immediate gains of their work, leaving them unsure of their impact within an organization. Feedback could be provided to workers in many forms such as summary data specific to that worker, summary information of their work compared to other workers, or in averages to let them know how well they are performing.

Though ODK was targeted specifically at organizations working in developing regions, there's no reason an organization could not publish publicly available forms and allow anyone with the application to submit data. Proper feedback would also likely be important to motivating more general community involvement for this type of crowd-sourced data collection [12]. However, this would also create new issues needing to be addressed involving data quality and privacy.

## *7.3 Summary*

Open Data Kit has enhanced organizations' ability to conduct data collection campaigns in developing regions by providing them with new and improved data collection techniques that allow them to collect and integrate information in ways previously not possible. However, we view our work as a stepping stone towards better, more powerful, more capable systems. Since technology by itself cannot force change, that is, human intent must be the driving force and technology can be the tool, ODK aims to help workers at all levels within campaigns with their tasks by making their jobs easier, more efficient, and more accurate. As technology advances we hope a

new generation of data collection tools will emerge from the foundations of our work. Though there still remains a long way to go towards simplifying the data collection process, the future remains wide open with possibilities.

### *7.4 Acknowledgements*

This research was made possible by a sabbatical at Google by Gaetano Borriello, and has received continued support from a Google Focused Research Award.

First, I would like to extend limitless thanks to my advisor, Gaetano Borriello, for inviting me to join him at Google to create the Open Data Kit project, and for his unwavering support and guidance. Thanks also to Richard Anderson, Noah Perin, and Kiersten Israel-Ballard at PATH for invaluable guidance in study design and implementation, and to Brittnay Fiore-Silfvast in the Department of Communications at UW for resuscitating our deployment. Finally, thanks to Dr. Sharad Iyengar, Dr. Kirti Iyengar, and Dr. Swati Gupta at ARTH who allowed me to spend a month with them learning their practices and implementing one of our studies.

Special thanks go to the other incredibly talented students who contributed greatly to ODK: Yaw Anokwa, Waylon Brunette, Adam Lerer, Clint Tseng; those who have supported, contributed to, and advised the ODK team: Julie Chin, Bill Schilit, Rebecca Moore, Sam Mbugua; and those who provided academic, moral, and writing support: Brian DeRenzi, Michael Buettner, and Ben Birnbaum. Special thanks also to all the organizations who have partnered with us and/or who have used our software and provided invaluable feedback: Grameen Foundation Applab, USAID-AMPATH, Dimagi, Jane Goodall Foundation, Surui Tribe, Vestergaard Frandsen and Manna Energy, Brazilian National Forest Service, Millennium Village Project, DataDyne, and the Berkeley Human Rights Center (now at Harvard). Our work only continues to be meaningful because these organizations continue to use and contribute to the software to help those in need.

# BIBLIOGRAPHY

[1] Apache license, 2.0. `http://www.apache.org/licenses/LICENSE-2.0.html`, January 2004.

[2] Jenny C Aker and Isaac M Mbiti. Mobile phones and economic development in africa. *Journal of Economic Perspectives*, 24(3):207–232, 2010.

[3] Amazon Simple Storage Service (S3). `http://aws.amazon.com/s3/`.

[4] AMPATH. IU-Kenya Partnership/AMPATH. `http://www.iukenya.org/hiv.aids.html`.

[5] Vishwanath Anantraman, Tarjei Mikkelsen, Reshma Khilnani, Vikram S Kumar, Alex Pentland, and Lucila Ohno-Machado. Open source handheld-based EMR for paramedics working in rural areas. *Proc AMIA Symp*, pages 12–6, Jan 2002.

[6] Richard Anderson, Chad Robertson, Esha Nabi, Urvashi Sahni, and Tanuja Setia. Facilitated video instruction in low resource schools. In *Proceedings of the Fifth International Conference on Information and Communication Technologies and Development*, ICTD '12, pages 2–12, New York, NY, USA, 2012. ACM.

[7] Yaw Anokwa, Carl Hartung, Waylon Brunette, Gaetano Borriello, and Adam Lerer. Open source data collection in the developing world. *Computer*, 42(10):97–99, October 2009.

[8] Yaw Anokwa, Nyoman Ribeka, Tapan Parikh, Gaetano Borriello, and Martin C. Were. Design of a phone-based clinical decision support system for resource-limited settings. In *Proceedings of the Fifth International Conference on Information and Communication Technologies and Development*, ICTD '12, pages 13–24, New York, NY, USA, 2012. ACM.

[9] Yaw Anokwa, Thomas N Smyth, Divya Ramachandran, Jahanzeb Sherwani, Yael Schwartzman, Rowena Luk, Melissa R Ho, Neema Moraveji, and Brian DeRenzi. Stories from the Field: Reflections on HCI4D Experiences. *ITID*, 5(4):101–116, Dec 2009.

[10] Action Research and Training for Health (ARTH). `http://arth.in/`.

[11] Christian Bauer, Max Rydahl Andersen, Emmanuel Bernard, Steve Ebersole, and Hardy Ferentschik. Hibernate reference documentation. *Significance*, 2007.

[12] Jeffrey Beorse, Yaw Anokwa, Carl Hartung, Waylon Brunette, and Gaetano Borriello. Dynamic data collection for participatory science in open data kit. In *Data Collection by the People for the People at CHI*, 2011.

[13] Matt Berg, Dr. James Wariero, and Vijay Modi. Every child counts the use of sms in kenya to support the community based management of acute malnutrition and malaria in children under five. Oct 2009.

[14] Benjamin Birnbaum, Brian DeRenzi, Abraham D. Flaxman, and Neal Lesh. Automated quality control for mobile data collection. In *Proceedings of the 2nd ACM Symposium on Computing for Development*, ACM DEV '12, pages 1:1–1:10, New York, NY, USA, 2012. ACM.

[15] J Blaya, W Gomez, P Rodriguez, and H Fraser. Cost and implementation analysis of a personal digital assistant system for laboratory data collection. *Int J Tuberc Lung Dis*, 12(8):921–7, Aug 2008.

[16] Joaquin Blaya and Hamish Fraser. Development, implementation and preliminary study of a PDA-based bacteriology collection system. In *American Medical Informatics Association Annual Symposium*, 2006.

[17] Joaquin Blaya and Hamish Fraser. Development, implementation and preliminary study of a pda-based tuberculosis result collection system. *AMIA Annual Symposium proceedings*, Jan 2006.

[18] Joaquin A Blaya, Hamish Fraser, and Brian Holt. E-health technologies show promise in developing countries. *Health Aff (Millwood)*, 29(2):244–51, Feb 2010.

[19] Eric Brewer, Michael Demmer, Melissa Ho, R.J. Honicky, Joyojeet Pal, Madelaine Plauch, and Sonesh Surana. The challenges of technology research for developing regions. In *IEEE Pervasive Computing*, volume 5, pages 15–23, Los Alamitos, CA, USA, 2006. IEEE Computer Society.

[20] Waylon Brunette, Wayne Gerard, Matthew A. Hicks, Alexis Hope, Mitchell Ishimitsu, Pratik Prasad, Ruth E. Anderson, Gaetano Borriello, Beth E. Kolko, and Robert Nathan. Portable antenatal ultrasound platform for village midwives. In *Proceedings of the First ACM Symposium on Computing for Development*, ACM DEV '10, pages 23:1–23:10, New York, NY, USA, 2010. ACM.

[21] Cell-Life. `http://www.cell-life.org/`.

[22] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly Detection: A Survey. *ACM Computing Surveys*, pages 1–72, 2009.

[23] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. Bigtable: a distributed storage system for structured data. In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation - Volume 7*, OSDI '06, pages 15–15, Berkeley, CA, USA, 2006. USENIX Association.

[24] Rohit Chaudhri, Waylon Brunette, Mayank Goel, Rita Sodt, Jaylen VanOrden, Michael Falcone, and Gaetano Borriello. Open data kit sensors: mobile data collection with wired and wireless sensors. In *Proceedings of the 2nd ACM Symposium on Computing for Development*, ACM DEV '12, pages 9:1–9:10, New York, NY, USA, 2012. ACM.

[25] Kuang Chen, Harr Chen, Neil Conway, Joseph M. Hellerstein, and Tapan S. Parikh. Usher: Improving data quality with dynamic forms. In *International Conference on Data Engineering (ICDE)*, 2010.

[26] Kuang Chen, Joseph M. Hellerstein, and Tapan S. Parikh. Data in the first mile. In *CIDR*, pages 203–206, 2011.

[27] Kuang Chen, Akshay Kannan, Yoriyasu Yano, Joseph M. Hellerstein, and Tapan S. Parikh. Shreddr: pipelined paper digitization for low-resource organizations. In *Proceedings of the 2nd ACM Symposium on Computing for Development*, ACM DEV '12, pages 3:1–3:10, New York, NY, USA, 2012. ACM.

[28] Shaohua Chen and Martin Ravallion. The developing world is poorer than we thought, but no less successful in the fight against poverty. *World Bank*, August 2008.

[29] Karen Cheng, Francisco Ernesto, and Khai Truong. Participant and interviewer attitudes toward handheld computers in the context of hiv/aids programs in sub-saharan africa. *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, Apr 2008.

[30] CommCare. `http://dimagi.com/commcare/`.

[31] CouchDB. `http://couchdb.apache.org`.

[32] Walter H Curioso, Bryant T Karras, Pablo E Campos, Clara Buendia, King K Holmes, and Ann Marie Kimball. Design and implementation of cell-preven: a real-time surveillance system for adverse events using cell phones in peru. *AMIA Annual Symposium proceedings / AMIA Symposium AMIA Symposium*, pages 176–80, Jan 2005.

[33] CyberTracker. `http://cybertracker.co.za/`, July 2010.

[34] DataDyne. `www.datadyne.org`.

[35] S. Day, P. Fayers, and D. Harvey. Double data entry: what value, what price? *Controlled Clinical Trials*, 1998.

[36] Nicola Dell, Nathan Breit, Timóteo Chaluco, Jessica Crawford, and Gaetano Borriello. Digitizing paper forms with mobile imaging technologies. In *Proceedings of the 2nd ACM Symposium on Computing for Development*, ACM DEV '12, pages 2:1–2:10, New York, NY, USA, 2012. ACM.

[37] Nicola Dell, Vidya Vaidyanathan, Indrani Medhi, Edward Cutrell, and William Thies. "Yours is Better!" Participat Response Bias in HCI. *CHI '12: SIGCHI conference on Human factors in computing systems*, May 2012.

[38] Brian Derenzi, Gaetano Borriello, Jonathan Jackson, Vikram S Kumar, Tapan S Parikh, Pushwaz Virk, and Neal Lesh. Mobile Phone Tools for Field-Based Health care Workers in Low-Income Countries. *The Mount Sinai journal of medicine, New York*, 78(3):406–18, May 2011.

[39] Brian DeRenzi, Krzysztof Z. Gajos, Tapan S. Parikh, Neal Lesh, Marc Mitchell, and Baetano Borriello. Opportunities for intelligent interfaces aiding healthcare in low-income countries. In *Workshop on Intelligent User Interfaces for Developing Regions (at IUI'08)*, 2008.

[40] Brian DeRenzi, Neal Lesh, Tapan Parikh, Clayton Sims, Werner Maokla, Mwajuma Chemba, Yuna Hamisi, David hellenberg, Marc Mitchell, and Gaetano Borriello. E-imci: improving pediatric health care in low-income countries. *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, Apr 2008.

[41] Lameck Diero, Joseph K Rotich, John Bii, Burke W Mamlin, Robert M Einterz, Irene Z Kalamai, and William M Tierney. A computer-based medical record system and personal digital assistants to assess and follow patients with respiratory tract infections visiting a rural kenyan health centre. *BMC Med Inform Decis Mak*, 6:21, Jan 2006.

[42] Dimagi. http://www.dimagi.com.

[43] Dropbox. http://www.dropbox.com/.

[44] eMOCHA Blog. http://emocha.org/blog/.

[45] Jodi L Vanden Eng, Adam Wolkon, Anatoly S Frolov, Dianne J Terlouw, M James Eliades, Kodjo Morgah, Vincent Takpa, Aboudou Dare, Yao K Sodahlon, Yao Doumanou, William A Hawley, and Allen W Hightower. Use of handheld computers with global positioning systems for probability sampling and data entry in household surveys. *Am J Trop Med Hyg*, 77(2):393–9, Aug 2007.

[46] Epihandy. http://epihandy.org/, July 2010.

[47] EpiSurveyor. http://www.episurveyor.org/.

[48] First Days. http://www.eecs.berkeley.edu/~divya/research/firstdays/.

[49] formhub. http://www.formhub.org.

[50] D Forster, R Behrens, and H Campbell. Evaluation of a computerized field data collection system for health surveys. *Bulletin of the World Health Organization*, Jan 1991.

[51] Jon Froehlich, Mike Y. Chen, Sunny Consolvo, Beverly Harrison, and James A. Landay. MyExperience: A system for in situ tracing and capturing of user feedback on mobile phones. In *MobiSys '07: Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*, pages 57–70, New York, NY, USA, 2007. ACM.

[52] Frontline Forms. http://www.frontlinesms.com/forms/.

[53] FrontlineSMS. http://frontlinesms.com/, July 2010.

[54] R Gandhi, R Veeraraghavan, and K Toyama. Digital green: Participatory video for agricultural extension. *International Conference on Information Technology and Communcation and Development*, Jan 2007.

[55] Google App Engine. https://developers.google.com/appengine/.

[56] Google SMS to Serve Needs of Poor In Uganda. http://blog.google.org/2009/06/google-sms-to-serve-needs-of-poor-in.html, June 2009.

[57] Grameen Foundation AppLab. `http://www.grameenfoundation.applab.org/`.

[58] GroupComplete. `http://www.groupcomplete.com`.

[59] D.L. Grover, M.T. King, and C.A. Kuschler. Reduced keyboard disambiguating computer. *Patent No. US 5818437 Tegic Communications, Inc. Seattle, WA*, 1998.

[60] Robert M. Groves, Floyd J. Fowler Jr., Mick P. Couper, James M. Lepkowski, Eleanor Singer, and Roger Tourangeau. *Survey Methodology*. Wiley-Interscience, 2004.

[61] Trish Groves. SatelLife: Getting relevant information to the developing world. In *BMJ*, 1996.

[62] Carl Hartung, Yaw Anokwa, Waylon Brunette, Adam Lerer, Clint Tseng, and Gaetano Borriello. Open Data Kit: Tools to Build Information Services for Developing Regions. In *ICTD '10*, 2010.

[63] Carl Hartung, Brian DeRenzi, and Gaetano Borriello. Decision support systems on mobile phones for low-income settings. In *International Mobile Health Workshop with 12th IEEE International Conference on E-Health Networking, Applications, and Service (IEEE Healthcom)*, July 2010.

[64] Carl Hartung, Richard Han, Carl Seielstad, and Saxon Holbrook. Firewxnet: a multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments. In *Proceedings of the 4th international conference on Mobile systems, applications and services*, MobiSys '06, pages 28–41, New York, NY, USA, 2006. ACM.

[65] F. Heider. *The psychology of interpersonal relations*. Wiley, New York, 1958.

[66] V J Hodge and J Austin. A Survey of Outlier Detection Methodoligies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

[67] Hae Sook Hong, Il Kon Kim, Sung Hee Lee, and Hwa Sun Kim. Adoption of a pda-based home hospice care system for cancer patients. *Comput Inform Nurs*, 27(6):365–71, Jan 2009.

[68] YoonSung Hong, Hilary K. Worden, and Gaetano Borriello. Odk tables: data organization and information services on a smartphone. In *Proceedings of the 5th ACM workshop on Networked systems for developing regions*, NSDR '11, pages 33–38, New York, NY, USA, 2011. ACM.

[69] Huawei ideos is top smart phone in kenya. `http://www.huawei.com/africa/en/catalog.do?id=761`, May 2011.

[70] International Telecommunication Union ICT Statistics. `http://itu.int/ITU-D/ict/statistics/`, July 2010.

[71] Inveneo. ICT project and sustainability primer. `http://inveneo.org/download/Inveneo_ICT-Sustainability_Primer0809.pdf`, July 2010.

[72] ITU. Information society statistical profiles 2009 - africa. *World Telecommunication Development Conference*, 2010.

[73] JavaRosa. `https://bitbucket.org/javarosa/javarosa/wiki/Home`, August 2010.

[74] Lois Jensen. *The Millennium Development Goals: Report 2010)*. United Nations Department of Economic and Social Affairs, New York, NY, USA, 2010.

[75] E E Jones. How do people perceive the causes of behavior? *American Scientist*, 64:300–305, 1976.

[76] Karin Källander, Helena Hildenwall, Peter Waiswa, Edward Galiwango, Stefan Peterson, and George Pariyo. Delayed care seeking for fatal pneumonia in children aged under five years in uganda: a case-series study. *Bull World Health Organ*, 86(5):332–8, May 2008.

[77] W Kaplan. Can the ubiquitous power of mobile phones be used to improve health outcomes in developing countries? *Globalization and Health*, Jan 2006.

[78] D. W. King and R. Lashley. A quantifiable alternative to double data entry. *Controlled Clinical Trials*, 2000.

[79] Ann E Kurth, Walter H Curioso, Elizabeth Ngugi, Lauren McClelland, Patricia Segura, Robinson Cabello, and Donna L Berry. Personal digital assistants for hiv treatment adherence, safer sex behavior support, and provider training in resource-constrained settings. *AMIA Annual Symposium proceedings / AMIA Symposium AMIA Symposium*, page 1018, Jan 2007.

[80] S O Lal, F W Smith, J P Davis, H Y Castro, D W Smith, D L Chinkes, and R E Barrow. Palm computer demonstrates a fast and accurate means of burn data collection. *J Burn Care Rehabil*, 21(6):559–61; discussion 558, Jan 2000.

[81] Jonathan Lester, Carl Hartung, Laura Pina, Ryan Libby, Gaetano Borriello, and Glen Duncan. Validated caloric expenditure estimation using a single body-worn sensor. In *Proceedings of the 11th international conference on Ubiquitous computing*, Ubicomp '09, pages 225–234, New York, NY, USA, 2009. ACM.

[82] Yen-Chiao Lu, Yan Xiao, Andrew Sears, and Julie A Jacko. A review and a framework of handheld computer adoption in healthcare. *Int J Med Inform*, 74(5):409–22, Jun 2005.

[83] Robert A Malkin. Design of health care technologies for the developing world. *Annu Rev Biomed Eng*, 9:567–87, Jan 2007.

[84] Burke W Mamlin, Paul G Biondich, Ben A Wolfe, Hamish Fraser, Darius Jazayeri, Christian Allen, Justin Miranda, and William M Tierney. Cooking up an open source emr for developing countries: Openmrs - a recipe for successful collaboration. *AMIA Annual Symposium proceedings AMIA Symposium AMIA Symposium*, 2006:529–533.

[85] Gayo Mhila, Brian DeRenzi, Caroline Mushi, Timothy Wakabi, Matt Steele, Prabhjot Dhaldialla, Drew Roos, Clayton Sims, Jonathan Jackson, and Neal Lesh. Using Mobile Applications for Community-based Social Support for Chronic Patients. *Helina*, 2009.

[86] Jakob Nielsen. Estimating the number of subjects needed for a thinking aloud test. *Int. J. Hum.-Comput. Stud.*, 41(3):385–397, 1994.

[87] Open Data Kit. `http://opendatakit.org/`, July 2010.

[88] OpenMRS. `http://openmrs.org/`, July 2010.

[89] OpenRosa Consortium. `http://openrosa.org/`, July 2010.

[90] OpenXData. `http://www.openxdata.org/`, July 2010.

[91] Orbeon forms. `http://www.orbeon.com/`.

[92] Tapan Parikh. Designing appropriate computing technologies for rural development (presentation). `http://people.ischool.berkeley.edu/~parikh/projects/CAM/papers/Appropriate_Computing.pdf`, May 2007.

[93] Tapan S. Parikh. *Designing an Architecture for Delivering Mobile Information Services to the Rural Developing World*. PhD thesis, University of Washington, 2007.

[94] Tapan S. Parikh. Engineering rural development. *Communications of the ACM*, 52(1):54–63, January 2009.

[95] Tapan S. Parikh, Paul Javid, Sasikumar K., Kaushik Ghosh, and Kentaro Toyama. Mobile phones and paper documents: Evaluating a new approach for capturing microfinance data in rural India. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 551–560, New York, NY, USA, 2006. ACM.

[96] Animesh Patcha and Jung-Min Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks*, 51(12):3448–3470, August 2007.

[97] PATH. `http://path.org/`.

[98] Somani Patnaik, Emma Brunskill, and William Thies. Evaluating the accuracy of data collection on mobile phones: a study of forms, sms, and voice. In *Proceedings of the 3rd international conference on Information and communication technologies and development*, ICTD'09, pages 74–84, Piscataway, NJ, USA, 2009. IEEE Press.

[99] Pendragon Forms. `http://pendragonsoftware.com/`, July 2010.

[100] Pendragon Forms Case Studies. `http://pendragonsoftware.com/casestudy/`, July 2010.

[101] M I Petrovskiy. Outlier Detection Algorithms in Data Mining. *Programming and Computer Software*, 29(4):228–237, December 2003.

[102] Zeshan Rajput, Sam Mbugua, David Amadi, Viola Chepng'eno, Jason Saleem, Yaw Anokwa, Carl Hartung, Gaetano Borriello, Burke Mamlin, Samson Ndege, and Martin Were. Evaluation of an android-based mhealth system for population surveillance in developing countries. In *American Medical Informatics Association (AMIA)*, 2011.

[103] Divya Ramachandran, John Canny, Prabhu Das, and Edward Cutrell. Mobileizing health workers in rural india. *CHI '10: Proceedings of the 28th international conference on Human factors in computing systems*, Apr 2010.

[104] RapidSMS. `http://www.rapidsms.org/`.

[105] Robin A. Reynolds-Haertle and Ruth McBride. Single vs. Double data entry in CAST. *Controlled Clinical Trials*, 13(6), 1992.

[106] Satellife. `http://healthnet.org/gather`.

[107] Cathy L Schell and Al Borchers. Codedoc for real-time point-of-care emergencies. *AMIA Annual Symposium proceedings / AMIA Symposium AMIA Symposium*, page 997, Jan 2003.

[108] Yael Schwartzman and Tapan Parikh. Using cam-equipped mobile phones for procurement and quality control at a rural coffee cooperative. *MobEA V: Mobile Web in the Developing World*, Jan 2007.

[109] Yael Schwartzman and Tapan S. Parikh. Establishing relationships for designing rural information systems. In *CHI '07 extended abstracts on Human factors in computing systems*, CHI EA '07, pages 1845–1850, New York, NY, USA, 2007. ACM.

[110] Christopher J. Seebregts, Burke W. Mamlin, Paul G. Biondich, Hamish S. F. Fraser, Benjamin A. Wolfe, Darius Jazayeri, Christian Allen, Justin Miranda, Elaine Baker, Nicholas Musinguzi, Daniel Kayiwa, Carl Fourie, Neal Lesh, Andrew Kanter, Constantin T. Yiannoutsos, and Christopher Bailey. The openmrs implementers network. *I. J. Medical Informatics*, 78(11):711–720, 2009.

[111] Jahanzeb Sherwani, Sooraj Palijo, Sarwat Mirza, Tanveer Ahmed, Nosheen Ali, and Roni Rosenfeld. Speech vs. touch-tone: telephony interfaces for information access by low literate users. *ICTD'09: Proceedings of the 3rd international conference on Information and communication technologies and development*, Apr 2009.

[112] Kizito Shirima, Oscar Mukasa, Joanna A. Schellenberg, Fatuma Manzi, Davis John, Adiel Mushi, Mwifadhi Mrisho, Marcel Tanner, Hassan Mshinda, and David Schellenberg. The use of personal digital assistants for data entry at the point of collection in a large household survey in southern Tanzania. In *Emerging Themes in Epidemiology*, volume 4, pages 5+, June 2007.

[113] Kizito Shirima, Oscar Mukasa, Joanna Armstrong Schellenberg, Fatuma Manzi, Davis John, Adiel Mushi, Mwifadhi Mrisho, Marcel Tanner, Hassan Mshinda, and David Schellenberg. The use of personal digital assistants for data entry at the point of collection in a large household survey in southern tanzania. *Emerging themes in epidemiology*, 4:5, Jan 2007.

[114] W. M. Tierney, J. K. Rotich, T. J. Hannan, A. M. Siika, P. G. Biondich, B. W. Mamlin, W. M. Nyandiko, S. Kimaiyo, K. Wools-Kaloustian, J. E. Sidle, C. Simiyu, E. Kigotho, B. Musick, J. J. Mamlin, and R. M. Einterz. The AMPATH medical record system: creating, implementing, and sustaining an electronic medical record system to support HIV/AIDS care in western Kenya. *Studies in health technology and informatics*, 129(Pt 1):372–376, 2007.

[115] Mark Tomlinson, Wesley Solomon, Yages Singh, Tanya Doherty, Mickey Chopra, Petrida Ijumba, Alexander C Tsai, and Debra Jackson. The use of mobile phones as a data collection tool: a report from a household survey in south africa. *BMC Med Inform Decis Mak*, 9:51, Jan 2009.

[116] Kentaro Toyama. Ten myths of ICT4D. `http://research.microsoft.com/en-us/um/people/toyama/talks/`, November 2009.

[117] Charles Tumwebaze and Frank Nkuyahaga. Epihandy mobile–a mobile data collection tool. *Proceedings of the 1st International Conference on M4D Mobile Communication Technology for Development*, pages 159–162, 2008.

[118] United Nations Department of Economic and Social Affairs, Statistics Division. *Household Sample Surveys in Developing and Transition Countries*, 2005.

[119] Jan Van den Broeck, Solveig Argeseanu Cunningham, Roger Eeckels, and Kobus Herbst. Data cleaning: detecting, diagnosing, and editing data abnormalities. *PLoS medicine*, 2(10):e267, Oct 2005.

[120] Rajesh Veeraraghavan, Naga Yasodhar, and Kentaro Toyama. Warana Unwired: Replacing PCs with mobile phones in a rural sugarcane cooperative. *ITID*, 5(1):81–95, Jan 2009.

[121] Martin Were, James Kariukic, Viola Chepng'enoc, Margaret Wandabwac, Samson Ndegec, Paula Braitsteina, Juddy Wachira, Sylvseter Kimaiyoc, and Burke Mamlin. Leapfrogging paper-based records using handheld technology: Experience from western kenya. In *Stud Health Technol Inform.*, pages 525–9, 2010.

[122] Martin C. Were, Burke W. Mamlin, William M. Tierney, Ben Wolfe, and Paul G. Biondich. Concept dictionary creation and maintenance under resource constraints: lessons from the AMPATH Medical Record System. *AMIA ... Annual Symposium proceedings / AMIA Symposium. AMIA Symposium*, pages 791–795, 2007.

[123] World Bank. *Moniotoring and Evaluation: Some Tools, Methods, and Approaches*, 2004.

[124] World Bank. *World Developent Indicators*, 2006.

[125] World Health Organization. *Integrated Management of Pregnancy and Childbirth*, 2010.

[126] XForms 1.1. `http://w3.org/TR/xforms/`, July 2010.

[127] ZXing, Barcode Scanning. `http://code.google.com/p/zxing/`.

# Appendix A

## USER EVENT LOGGING FOR ARTH DEPLOYMENT

The following is a complete list of user interactions and events that we logged in the ARTH Deployment.

- Booting the phone
- Shutting down the phone
- Changing the time
- Changing the date
- Changing the time zone
- Starting a new form
- Loading a saved form
- Stopping a form mid form and saving it
- Stopping a form mid form and not saving it
- Saving a form at the last screen
- Exiting the program
- Swiping forward to the next question

- Swiping back to the previous question
- Swiping forward and encountering a constraint violation
- Using the hierarchy menu to jump to a new question
- Clearing an answer
- When a video is started
- When a video is stopped
- When the breath counter button was pressed
- When the stopwatch start/stop/reset buttons were pressed
- When a sync was attempted