

Open Data Kit

Building Information Services for Low Income Regions

Yaw Anokwa, Carl Hartung, Waylon Brunette,
Adam Lerer, Clint Tseng, Gaetano Borriello

<http://opendatakit.org>

My name is Yaw Anokwa. I'm a Ph.D. student in computer science at the University of Washington. Today's talk is about Open Data Kit, or ODK.

ODK is a set of open source tools for building information services. It's been designed to work in low income regions and the idea is that organizations can pick and choose the tools they need to build their particular information service.

This is work that is done with Carl Hartung, Waylon Brunette, Adam Lerer, Clint Tseng, and Gaetano Borriello. Open Data Kit is an open source project and you can find out more at <http://opendatakit.org>

This presentation is based on a paper that will appear at ICTD 2010 in December. That paper has a lot of detail, so for this talk, I'll focus on a broad overview and touch on some of the interesting things we've learned.

I'll start with some background and motivation. I will then go over a few of the tools our team has built and end with some qualitative feedback from our users.

I'll use that feedback to convince you that the design decisions we have made in ODK enables functionality for building information services that hasn't existed before.

If you have questions, feel free to ask. We will also have some time at the end of the talk.

So what is an information service?

Inshuti Mu Buzima
Clinique des Maladies Chroniques

Patient ID: - **DONNÉES DE BASE**

1. Date d'anniversaire:	2. Nom:	3. Sexe: <input checked="" type="checkbox"/> F <input type="checkbox"/> M	4. (a) Date de naissance:	Prénom:
5. Lieu de naissance:	6. Pour les enfants: (a) Personne responsable: Nom:	(b) Lien avec le patient:	(b) Age: 35 ans mois	
7. Résidence actuelle: Province:	8. Le patient a été référé en provenance de:	Cellule:	Prénom:	
Secteur:	(a) Urgence <input type="checkbox"/>	(b) Clinique ID <input type="checkbox"/>	District administratif:	
9. Catégorie du patient:	(c) Maladie du cœur <input type="checkbox"/>	(d) Soin primaire <input type="checkbox"/>	Umuudugudu:	
10. Groupe du patient:	(e) Désordre de saisisse <input type="checkbox"/>	(f) Maladie des reins <input type="checkbox"/>	Autres:	
11. Clinique:	(g) Asthme <input type="checkbox"/>	(h) Diabète <input type="checkbox"/>		
(a) Rusumo <input type="checkbox"/>	(b) Autre (à spécifier) <input type="checkbox"/>	(c) Mulindi <input type="checkbox"/>		
Information de la communauté				
12. Le patient habite près de quel centre de santé:				
13. Est-ce que le patient possède un accompagnateur:				
Si oui: Nom de l'Accompagnateur:				
Si non: (a) Nom de l'agent de santé communautaire: Nom: _____ (b) Accompagnateur <input type="checkbox"/> animateur de si: _____				

Inshuti Mu Buzima

Amazina:
 Clinique: CCC
 Umuherekeza: _____
 Patient ID N°: _____
 Date de début: _____

(1) amotung X (2y.)

Pour la maladie chronique, indiquer les médicaments(s)

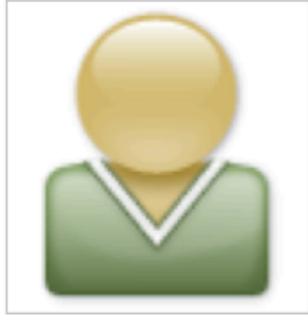
I work mostly in healthcare in sub-Saharan Africa, so let me give you an example of what I mean in that domain.

When a patient comes into a clinic, the nurse fills out a paper form about the visit. The paper form goes into the patient's folder, and then when the patient comes back a few months later, the doctor reviews what the folder and maybe along with some lab results written on paper can make a decision.

So the data entry is done on paper, and the retrieval of that data is also done on paper. I call this a paper-based information service.

With a paper-based healthcare system like this becomes really easy to miss trends in a record, misread someone's writing and prescribe the wrong drug. It's also hard to search or transport this paper.

Ideally, you want to make this system more computerized so you can address some of those issues.



Paul Persil Patient

[Back](#)[Print](#)Gender **Male**Age **44 years** (~ Jun 01, 1934)Last Visit **1 week ago** (Aug 14, 2008)

Doug Doctor, Rwinkwau Hospital

TRACnet ID: 12345

Carte d'Identité: 1234567

IMB ID **12345678-**
A

Alerts

NO XRAY RESULT IN THE LAST 6 MONTHS**NO CXR RESULT**

Notes

No known allergies

Recent Symptoms

Hypersensitivity/allergic reactions, purple toes syndrome

Date

Aug 14, 2008

Anemia, pallor, fever, rash, dermatitis, including bullous eruptions

Jul 01, 2008

Dizziness, loss of consciousness, taste perversion, and alopecia

Feb 27, 2008

Drug Order

Dose

Frequency

Start Date

Stop Date

Notes

AZT+3TC

1.0 tab(s)

2/d x 7 d/w

Sep 13, 2007

D4T

150 mg

1/d x 7 d/w

Aug 02, 2007

Triomune-40 (stopped)

1.0 tab(s)

2/d x 7 d/w

Sep 01, 2007

Sep 13, 2007

Unexplained facial rash

EFV 600 (stopped)

1.0 mg

1/d x 7 d/w

Aug 02, 2007

Sep 13, 2007

Lab Test

Result

Date

Notes

CD4**512**

Aug 15, 2008

Ordered by Dr. Doctor

CD4**259**

Aug 01, 2008

Viral load**515**

Jul 28, 2008

Second test for verification of status

Viral load

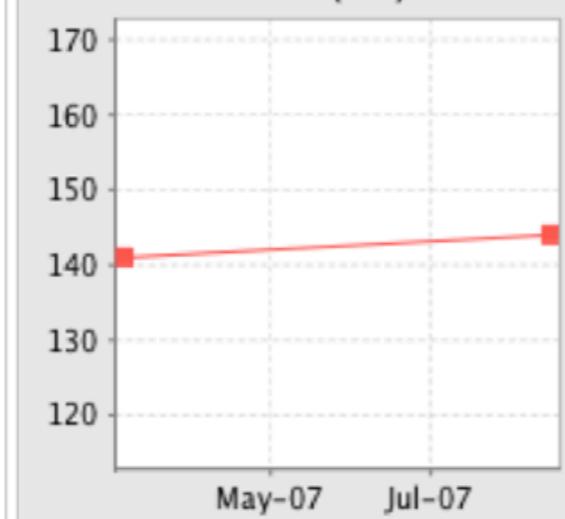
200

Jul 27, 2008

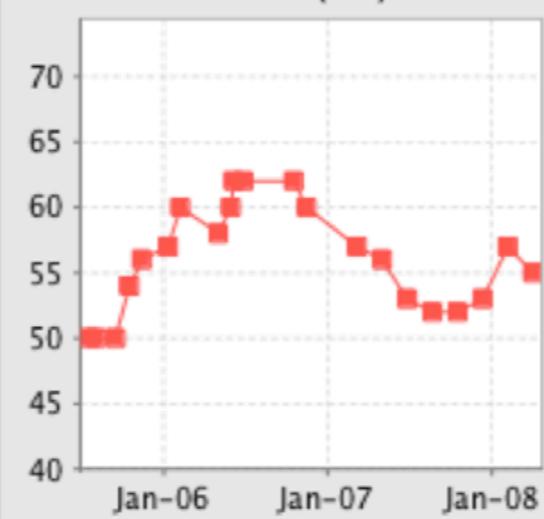
Ordered by Dr. Green

Graphs

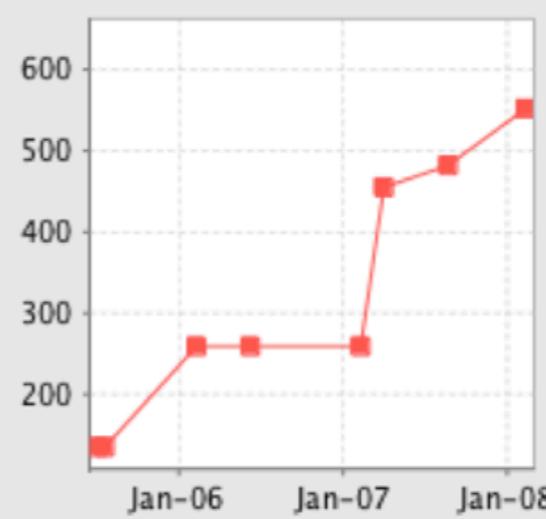
HEIGHT (CM)



WEIGHT (KG)



CD4 COUNT



This is a summary of an HIV patient's record generated from the OpenMRS medical record system.

You enter the patient's information using a phone, tablet or computer. Test results from the lab automatically appear in the patient's record. Graphs of weight are generated on demand. Algorithms can find problems in the record and alert clinicians about mistakes.

In this case, we've taken paper-based information service and converted it to an computerized information service.

Beyond the medical domain there are other paper-based systems you might want to consider making more computerized...

- Government workers completing surveys about households in a district.
- Microfinance institution tracking transactions from lenders and borrowers.
- Crisis mapper tasked to capture images and locations of damaged areas after a hurricane.

If you are doing a socio-economic survey of a remote region, it'd be great if you didn't have to transport all your paper forms via Land Rover and waste all that fuel. You might want to send those results over the cell network so they appear at the main office. You might even want to make the forms smart -- make sure ages are between 0 and 100 or that men don't get asked questions about pregnancy.

If you are doing microfinance and dealing with a lot of numbers. You might want to make sure the data being entered in the paper form all add up, before giving out the loan. You might want to send the recipient an SMS a week before their loan is due as a reminder.

There are also more complex examples where ordinary paper-based systems just doesn't work.

If there is a hurricane and buildings have been destroyed, it'd be good to have pictures and GPS coordinates of those buildings. It'd also be good if you could do this in real time and task people to go to specific sites.

Basically, our claim is...

Paper-based practice in low-income regions limits the scale and complexity of the information services that can be provided and thus the impact of the intervention.

We aren't the only people who have had this idea. With the growth of cell phone usage all over the world, there have been lots of projects where people use phones and computers to replace paper.

Let me touch on why the current approaches are lacking.



Staying with the medical domain, here is a picture of a nurse using a PDA to do some decision support. This picture is a good example of everything that is wrong with the current approaches.

First, there is the choice of the platform. A lot of projects use PDAs and basic Java phones. Much of this is cost driven -- you want the cheapest device possible.

The problem with this approach is that it traps your application on platforms that don't last very long. How many of you are still using a Palm Pilot? Anyone know a manufacturer that is continuing to ship Windows Mobile phones?

The cheapest device possible also traps your application on a platforms that make it hard to innovate. On basic feature phones, it's really hard to do simple things like take a picture or have a complex database without spending lots of time and money. The apps also tend not to work across all the different kinds of phones people have.

Finally, there is also the cost of training. Teaching someone to use a stylus or entering a long patient name with a basic phone keyboard is really hard -- you spend a lot of time and money doing that. In low income regions, usability matters.

- Cost-cutting at the device level leads to deploying on dying platforms that are hard to program and hard to train for.
- There are also challenges in how extensible a platform is across domains and how quickly you can move data across systems.
- How do you enable this system to scale without input from the original designers? Can we lower the bar to independent innovation?

So we've talked about cost-cutting at the device level. Projects like EpiSurveyor and FrontlineForms suffer from this. They are great for basic applications, but it's hard to innovate, hard to train and the device lifetime is really short.

There is also how quickly you can extend the platform across domains. Why can't we build a system that works for healthcare and microfinance and crisis mapping? More importantly, why can't the data we need move across other systems? Pendragon Forms has this problem, you can't easily move your data around.

Finally, there is how you grow and support this community as the project leaders leave the project. Can we put these services on the technology curve which can drive the cost down, but still allow for innovation? CAM is a project from a few years back that did a lot of things correct, but failed to build a lasting community.

These are the issues we tried to solve when building ODK. So with that, let's take a look at what some of the tools actually are.

ODK Collect

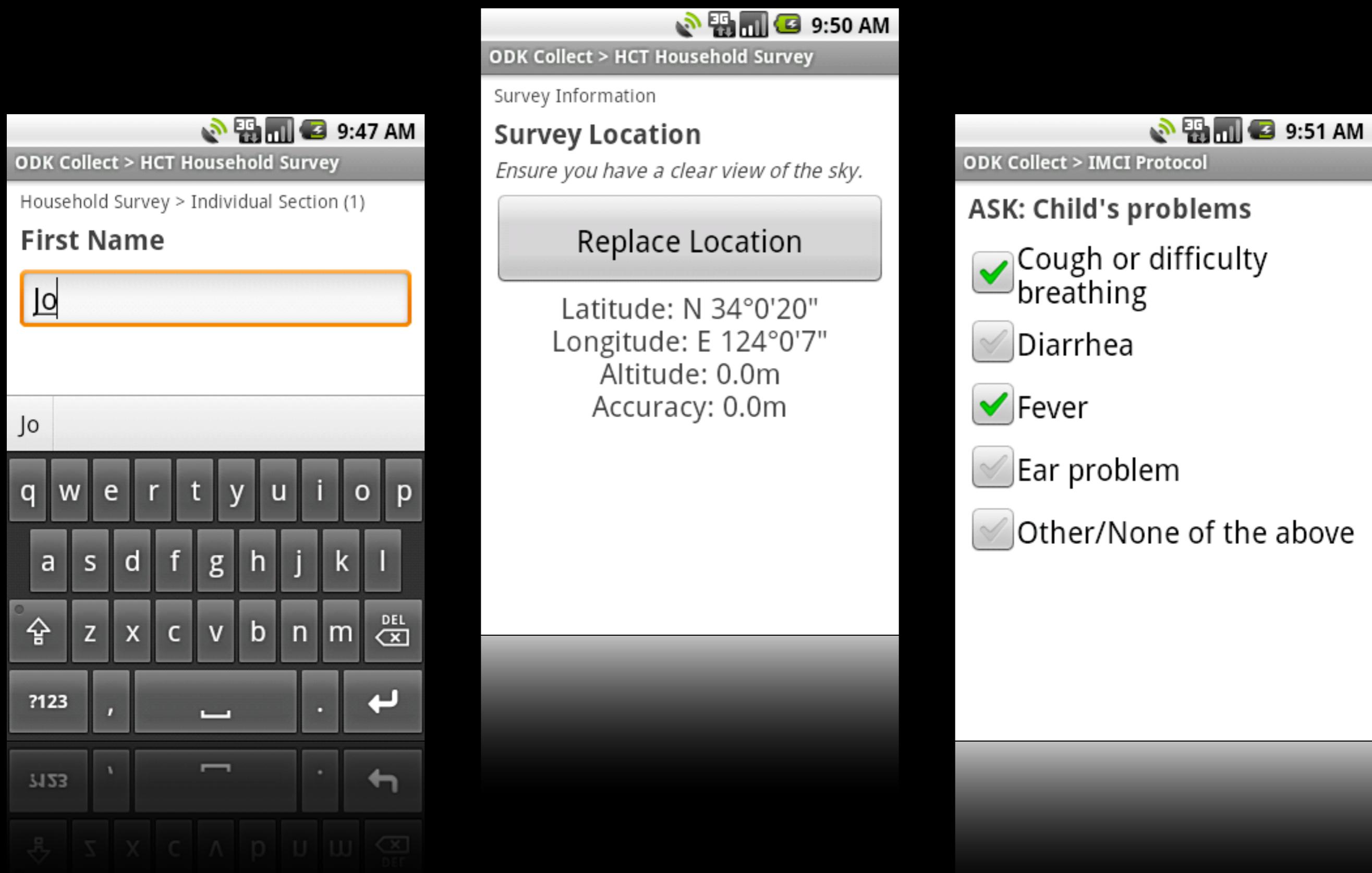
Mobile-based data collection and delivery



Our first and most popular tool is ODK Collect, which is a mobile-based data collection and delivery system.

This is a picture of a community health worker in Kenya entering data on a Google's G1 smart phone -- the first Android device. It runs an advanced (and open source) operating system. It has a touch screen, 3G and WiFi, GPS, great camera, full keyboard, etc. It's basically a computer.

The way Collect works is that users get a prompt-at-a-time when collecting or delivering information. They go forward and backward by using a swipe motion moving their finger across the screen (similar to turning a page in a book).



Collect's prompts include standard data types like text and numbers. It also supports advanced data types including GPS, pictures, audio, video, barcodes, etc. They can also be in multiple languages, they can be grouped, and they can loop.

The prompts are smart. They can have constraints so you always enter data that is valid. For example, it can check to make sure age isn't greater than 100.

Collect can also give prompts based on previous answers. This logic and the multimedia can be paired to deliver information.

For example, if you are coughing and have a fever, we can show you a video about coughing and fever and recommend medication. If you say diarrhea, then we can show you a video about that and then give you information about safe water practices.



Collect also makes it easy for others to build extensions.

In this case, instead of the health worker typing in lot of text, she can use the camera to scan a barcode which has that same information. You never have to type a really long id number and so you get less errors.

Someone else wrote the barcode scanner program and this is the kind of innovation that is possible with Collect. If you have the scanner program installed, you can use it. If not, you get a regular text box.

You can do similar things with other sensors or other applications. So for example, if you have a full medical record system on the phone, Collect could pull that information into a session.

- Collect uses intelligent multi-lingual prompts to collect and deliver information using a wide variety of textual and multimedia formats.
- Logic and data is stored as XML and can be transferred asynchronously via simple interfaces.
- Android design allows external developers to rapidly add functionality without touching core of ODK Collect.
- Works on a wide range of devices from smartphones to netbooks to tablets.

So where does the logic come from and the data where does it go?

ODK Aggregate

Scalable data storage and transfer

	Location-Longitude	Location-Altitude	Location-Accuracy	Description
64	-83.68864473333333	300.6	4.0	Mtri drain north
783	36.81718647480011	1675.0	6.0	Icdfut
	null	null	null	Testing in kampala
				table
				null
				kuca
				What?
				DevSeed Office
				teste
				Ethan's party
				Test
				Disnsjdv
				Test image
				Mathieu
	null	null		desk muc
3	-71.42707049846649	4.0	6.0	Computer in my room.
2	-71.42747819423676	80.0	96.0	Home
2	-120.4217004776001	72.0	384.0	Chris
625	175.61919629573822	63.0	4.0	Massesy carpark outside Landcare
625	175.61919629573822	63.0	4.0	Massesy carpark outside Landcare

Aggregate that hosts prompt logic and submitted data and provides interfaces for extraction such as spreadsheets, maps, and queries.

It is an scalable level storage and transfer and was designed to run on the Internet as a cloud service. So in the same way that email services run in the cloud, we wanted Aggregate to do the same things and lower the bar for deployment.

The key here is that there are cloud services (namely Google's App Engine, Amazon's Web Services, Microsoft's Azure) free or cheap and take care of all the security and maintenance and upgrades and scale issues. Aggregate makes that server piece easy for non-geeks to deploy and tends to be more reliable than running a server in Africa.

Of course, because we know people work in disconnected environments or want to control their data so, you can choose to run it locally. The same code base runs on a local Tomcat server backed with MySQL or Postgres

The screenshot shows a mobile application interface. On the left, there's a map of a forested area with a large cluster of yellow pushpins marking specific locations. A callout bubble points from one of these pins to a detailed view of a tree trunk and branches on the right. Below this image is a table of data fields and their values.

SubmissionDate	null
DeviceId	351676030226551
SurveyorName	
TreeLocation-Latitude	-4.919064044952393
TreeLocation-Longitude	29.60844397544861
TreeDBH	45.0
TreeName	null
TreePicture	View

Aggregate doesn't have to hold your data forever. It can export data to other systems.

In this example, some forestry workers in Tanzania submitted data from Collect to Aggregate and then exported to Google Earth. Managers can click on each yellow point and get the data that was submitted.

Data from Aggregate can also go to SalesForce or Drupal or Google Spreadsheets in real time. To do this, you just add code that connects to that service.

- Aggregate provides a server repository for prompt logic and submitted data. Also provides open interfaces to extract data and integrate with existing systems.
- Designed to run on both cloud infrastructure or on local infrastructure. One codebase that can run on Google App Engine, Amazon Web Services, and Tomcat.
- Abstracts the difficulties of relational databases and simplifies extraction by using queries, maps, reports, etc.

If you wanted to put together an information service, how would this work?

ODK Build

Design services with drag and drop

ABC First name fname

Please record your location location

When is your birthday? bday

Please take a picture of yourself image

Properties

Data Name

The data name of this field in the final exported XML.

 image

Caption Text

The name of this field as it is presented to the user.

 English

Please take a picture of yourself

Hint

Additional help for this question.

 English Read Only

Whether this field can be edited by the end user.

 Required

Whether this field must be filled in before continuing.

Kind

Type of media to upload.

Advanced

[+ Add new](#)

Text

Numeric

Date

Location

Media

Choose One

Select Multiple

Build is an HTML5 application called designers can use. It runs in the browser but could also run offline.

Let's say you want to build a name and location survey. You drag and drop each prompt the user will interact with from this button pane to the canvas. Each prompt has a set of properties which users can edit. Users can rearrange ordering or add custom logic as needed.

Build generates an XML file that is the heart of ODK.

```

<?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>Geo Tagger v2</h:title>
    <model>
      <instance>
        <geotagger id="geo_tagger_v2" >
          <DeviceId/>
          <Image/>
          <Location/>
          <Description/>
        </geotagger>
      </instance>
      <bind nodeset="/geotagger/Device" type="string"
            | jr:preload="property" jr:preloadParams="deviceid"/>
      <bind nodeset="/geotagger/Image" type="binary"/>
      <bind nodeset="/geotagger/Location" type="geopoint"/>
      <bind nodeset="/geotagger/Description" type="string"/>
    </model>
  </h:head>
  <h:body>
    <upload ref="Image" mediatype="image/*">
      <label>Capture the image.</label>
    </upload>
    <input ref="Location">
      <label>Capture the location.</label>
    </input>
    <input ref="Description">
      <label>Describe the image and location.</label>
    </input>
  </h:body>
</h:html>

```

XForms are an XML-based form description and data exchange standard designed by the W3C as the next generation of web forms. It's an open standard and it's Turing complete.

Supports a wide variety of control and data types including: text, integer, decimal, select-one, select-multi, image, audio, video, barcode, and location. Also includes entry constraints, read-only prompts, required fields, multi-lingual translations, and branching logic.

We use XForms to describe the application logic for Collect, but also to generate the storage logic for Aggregate.

You can upload that the form to Aggregate and Collect can download the XML file that gives you the prompts you see. That same form on Aggregate also builds the specific database that Collect can submit to.

```
<instance>
  <geotagger id="geo_tagger_v2" >
    <DeviceId>00808009845125342</DeviceId>
    <Image>12315123213.jpg</Image>
    <Location>10.12304017412,-123.12309712402197</Location>
    <Description>A picture of me in Seattle</Description>
  </geotagger>
</instance>
```

When you finish using Collect, it puts the data in this upper part of the form and sends just this.

Because data like this is an open standard, it allows us to exchange data with other XForms compatible systems. It also makes it really easy for others to connect to us.

- Build makes simple drag and drop information services using a web based UI.
- XForms is an open standard that provides the application and server logic as well as the data exchange format for ODK.
- XForms ecosystem enables compatibility with a wide variety of non-ODK tools. Interactions with existing or future systems become possible.

Collect	Mobile-based data collection and delivery
Aggregate	Scalable data storage and transfer
Build	Design services with drag and drop
Voice	IVR-based collection and delivery client
Dropbox	Lightweight file store for desktops
Tasks	Task mobile workers from servers
Manage	Automate application and data delivery
Visualize	Visualize collected data on server
Clinic	Capture and view clinical data on mobile

We've covered Collect, Aggregate and Build. ODK is a more than these three tools and I want to touch on some the other tools we have.

ODK Voice is an IVR-based client. It does all the things Collect does, but it does it using the voice channel. For example, you can have Voice call all the people in a district and ask them for their age, gender. Based on their responses with their keypad you can play back age and gender specific advice.

The respondents push the buttons on their basic phone to get or give information and all the data gets fed back to Aggregate. So again, it's just like Collect, it uses the exact same XForm and data exchange, but it works with the broader population.

Dropbox is an example of a lightweight server. If you are running a small field deployment, you might not want to setup something like Aggregate. Dropbox is really basic and just gives you the raw submission files.

Collect can submit to Aggregate or Dropbox and Voice can submit to Aggregate. This is what we mean by being able to pick and choose the tools you need for your deployment.

We are currently working on a bunch more tools like Tasks, Manage, Visualize and Clinic.

- ODK is designed to be composable and usable. You pick and choose the configuration you like.
- ODK is designed to work with non-ODK systems. You use simple interfaces to exchange logic and data.
- ODK is built on open source and has a strong community behind the software and the hardware. You are free to do what you want.

These are nice bullet points but is there any evidence that ODK does the things it claims?

We selected four groups who were familiar with existing systems and had used ODK for some time. We asked these groups to describe how they use the platform, what alternatives they considered, what capabilities ODK enabled.

Here is what they said.



“It is relatively easy to train enumerators using ODK as the interface is highly usable, even for people who have little or no computer experience.”

Berkeley Human Rights Center

HRC documents human rights violations. They fly into a country, very quickly train data collectors and have to do very large surveys. They tried EpiSurveyor but found the cost and limited number of available devices was a problem.

They now use ODK in Uganda and Central African Republic and have collected thousands of hours worth of surveys. They have also made their own ODK compatible tools and released those into the public. For them ease of use was really important.

“We have found that the ability to synchronize and analyze data daily, as it comes in significantly improves the quality of the data.”

Berkeley Human Rights Center

These guys aren't really programmers, but they were able to build their own tool called Kobo that helped them analyze their data just in time and make adjustments daily.

“Before using ODK, we had created data collection functionality on Palm TX devices using Pendragon Forms. GPS information was collected using eTrex devices.”



AMPATH

AMPATH is a large hospital network in Kenya trying to counsel and test about 2 million people for HIV.

Previous to ODK, they used Pendragon Forms with PDAs and external GPS but found challenges with training. For example, plugging in the GPS via cable and typing in the coordinates twice to make sure you got no errors took a lot of training.

It was also hard to integrate with their existing medical record system or make modifications to the mobile client.

ODK solved problems with the “*clumsy and unreliable*” PDA and GPS. Training was “*much easier*” and the captured data suggests fewer data entry errors.

AMPATH

With ODK, getting GPS is a button press. They could also submit their data directly to OpenMRS instantly instead of syncing their PDAs to MS Access and then transferring that data over occasionally.

AMPATH switched to ODK last year have a few hundred Android devices being used in Kenya. They are running studies right now but their initial numbers show cost savings.

“Features sometimes trump cost, one has to make the choice between enhanced UI vs minimal UI, advanced features vs basic functionality, in our case we needed to get more out of the phone and ODK allowed us to do that.”

D-Tree International

D-Tree is an NGO that uses PDAs and basic phones to do clinical decision support primarily in Tanzania. They've always used the cheapest phones, but have been trying to put more medical data on phones.

D-Tree has created an peer-to-peer application that synchronizes patient information across other phones and servers. This functionality enables portable patient record systems that works across a variety of connectivity scenarios found in developing regions.

Collect can integrate with this system.

“Quick and easy user interaction...a bit of technophobia by users when it comes to the higher end phones that ODK runs on.”

D-Tree International

At the same time, D-Tree had problems. They've been concerned with security issues of the smart phones being used in really poor areas. For this reason, they are deploying ODK in facilities.

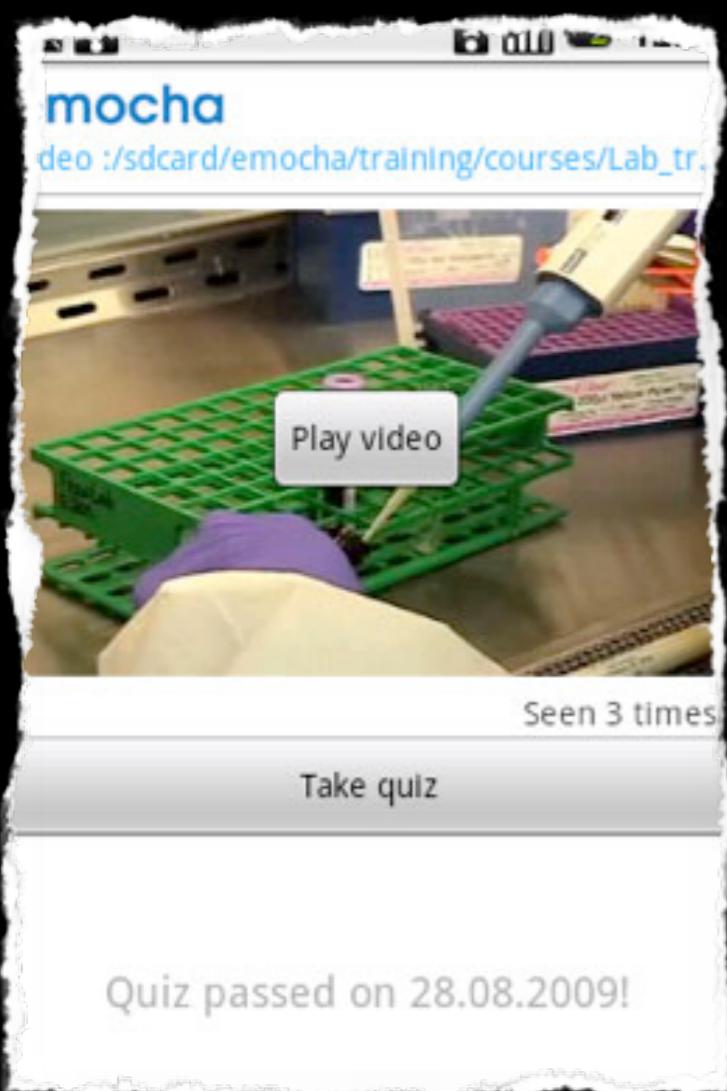
“...we did not find other open source alternatives we could integrate. We did consider creating a form system ourselves. We chose to go for ODK to save development time and follow the XForms standard, even we would lose freedom and flexibility.”

John’s Hopkins Center for CGHE

John’s Hopkins wanted to build a system that improved health care provider communication and information.

They have their own tools, like a server that generates clinical statistics and a phone client that has lots of video courses. They needed a way to test their users and so that’s what they use ODK for.

Like all programmers, they did struggle with losing the ability to do whatever you want with a platform -- working in a community is sometimes hard.



“[ODK] is used for patient data gathering, and to give an exam after watching a video course on the phone, to make sure the user understood the content.”

John's Hopkins Center for CGHE

Despite the initial concerns, they've been pretty happy.

- HRC: ODK's cost of deployment and short timelines drove much of their decision.
- D-Tree: Impetus was backwards compatibility paired with ease of use and development.
- JHCCGHE: Focused on integration with a larger set of mobile and server tools.
- AMPATH: Needed to integrate with a medical record system and usability was key.

So what we conclude is this...

Based on the feedback from four implementers, ODK has demonstrated it can enhance information services in a variety of low-resource environments.

Our users report ODK is easier to use, more capable, more cost-effective, and more accurate than the alternatives they considered.

The number of users we asked is small but we also have a steady growth of groups picking up ODK and using it -- not just in developing countries but in developed countries as well.

We have seen non-experts build working information services.
We have seen independent developers check in code.
We are seeing companies making money building services with the platform.
We are also seeing previous systems like CyberTracker, MyExperience and EpiSurveyor are also porting their work to ODK.

These anecdotes are not evidence, but they suggest we are on the right path.

Going forward we need to prove if the relatively higher cost of the phones cover the anecdotes of cost savings from less training and data accuracy.

Building more tools that exploit the smartphone functionality. Examples include ultrasound on the phone, richer text free interface, workflow management, etc.

So that's the end of my talk.

Before I take questions, I need to make a plug for Change.



ODK is the result of Change – a group based at the University of Washington exploring how technology can improve the lives of underserved populations.

<http://opendatakit.org>

<http://change.washington.edu>