Helping the Helpers: A Toolkit for Mobile Humanitarian Assistance Apps

by

WeiHua James Li

S.B., Massachusetts Institute of Technology (2014)

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of Master of Engineering in Electrical Engineering and Computer Science at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2016

© Massachusetts Institute of Technology 2016. All rights reserved.

Author .... Signature redacted ....

Department of Electrical Engineering and Computer Science

January 4, 2016

Accepted by .... Signature redacted ....

Dr. Christopher J. Terman

Chairman, Master of Engineering Thesis Committee

Certified by .... Signature redacted ....

Lalana Kagal

Principal Research Scientist

Thesis Supervisor

Certified by ... Signature redacted ....

Hal Abelson

Class of 1992 Professor of Computer Science and Engineering

Thesis Supervisor

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

JUL 19 2016

LIBRARIES

ARCHIVES

Helping the Helpers: A Toolkit for Mobile Humanitarian Assistance Apps

by

WeiHua James Li

Submitted to the Department of Electrical Engineering and Computer Science
on January 4, 2016, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

My research investigates the following question - How can relief workers be empowered to create useful mobile apps to support the work of first responder teams? Mobile devices are reshaping the disaster management domain and they make fast and targeted support possible. However there is the major obstacle of relief workers and volunteers often lacking the technical abilities to build and deploy the right mobile app in response to a particular disaster. In addition, data interoperability is often missing in many of the applications since they were developed without any prior agreement on the data schema. Linked Data technology solves the data interoperability problem by defining a method of publishing structured data and these structured data can be interlinked and become more useful. In order to explore and provide a solution to the gap between the knowledge of the relief workers and volunteers and the technical abilities needed to create an app, I conducted a participatory design workshop with the people at the International Committee of Red Cross and developed two mobile applications with one of the project managers there. In addition, I have created a Do-It-Yourself (DIY) toolkit that includes: (1) a Mobile Linked Data App Kit: a streamlined process for creating mobile apps and (2) an app-building methodology: a set of principles for relief workers to follow while creating apps using the toolkit.

The Mobile Linked Data App Kit has the Punya framework, the Linked Data Form Extension, and the Virtuoso Docker container. The Punya framework is a DIY app-building platform utilizing drag-drop visual blocks to create new mobile applications. It includes Linked

Data technology. The framework supports many Linked Data features. One of them is the ability to interoperate with different data sources needed for humanitarian assistance. Linked Data form is a critical part of data interoperability in the Linked Data technology. The Linked Data Form Extension is a tool implemented as an extension to the Punya framework that automatically generates Linked Data forms for Punya mobile apps.

Linked Data needs to be saved at the triplestore and Virtuoso is one of the leading providers for Linked Data triplestore. The Virtuoso Docker container makes it possible to set up an instance of the Virtuoso Linked Data store in minutes with a few simple commands. These automated steps relieve developers the burden of manually creating them. The app-building methodology is for relief workers to follow while creating mobile apps.

The goal of this toolkit is to enable aid workers to create vital and necessary mobile apps and to empower the humanitarian community. User testing of the framework with first responders has shown that the idea of "using, modifying, and creating" an app is greatly favored by our participants, that mobile Linked Data applications can aid humanitarian organizations by increasing their impact and effectiveness, and that humanitarian workers can successfully develop and deploy mobile applications by using this toolkit.

Thesis Supervisor: Hal Abelson
Title: Class of 1922 Professor of Computer Science and Engineering
MIT Department of Electrical Engineering and Computer Science

Thesis Supervisor: Lalana Kagal
Title: Principal Research Scientist
MIT Computer Science and Artificial Intelligence Laboratory

Acknowledgments

Foremost, I would like to thank my research advisor Hal Abelson for his immense support, patience, and guidance in helping me complete this thesis. Hal has tirelessly motivated me throughout my undergraduate and graduate career at MIT. Hal's enthusiasm and dedication in pursuing research about democratization of technology and information has propelled me into my graduate study and has made a significant impact on my career.

I would also like to thank my thesis advisor Lalana Kagal for her strong support and guidance in the pursuit of this thesis. I could not have done this without her help. Lalana listened to my research patiently and encouraged me to explore on my own.

I want to express my gratitude to my groupmates at App Inventor team and Decentralized Information Group for their support throughout the past five years. A huge thanks to Fuming Shih who taught me how to conduct research and always listen patiently to my research problems. Thanks to Ilaria Liccardi who spent a lot of time in proofreading my thesis and providing guidance for my thesis. To Jose Dominguez, who spent endless hours helping me on App Inventor development. To Marisol Diaz, who provided so much administrative support to me during my undergraduate and graduate career at MIT.

I am thankful to many friends I have met at the Church in Cambridge. Thank you for keeping me in your prayers and your endless support. Your companionship has made my last two years at MIT much more bearable and enjoyable. The Monday dinner at the MIT dormitory is one of my landmark experiences.

Finally, I want to thank my parents and sister, for the countless moral support and assistance they have provided me throughout college. They never fail to believe in me. Thank you very much for making yourself available whenever I needed someone to talk to. You are the fuel for my determination to finish my studies at MIT and I am greatly indebted to you all.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1. 1 Overview

Over the last few decades, mobile phones have evolved from just being portable communication tools into serving as high-powered information centers. They are mobile computing hubs that are deeply integrated into almost all aspects of people's daily lives. Mobile devices are used to request and pay for taxis, they are used by students to submit their homework and used as personal navigation aids by tourists. This is in addition to being used for reading news, taking pictures and watching videos. Yet despite the widespread use of mobile devices, humanitarian organizations are often limited in their in their ability to make effective use of mobile computing. NGOs and humanitarian organizations face issues related to data collection and data management, and first responders will often resort to slow manual work instead of potentially more efficient automatic processes due to a lack of appropriate mobile computing tools.

That being said, the ubiquity of handheld computing technology has become invaluable to disaster management and relief operations [23]. Data collection and management through handheld devices accelerates the aggregation and usability of data, with the ability to connect to many other data sources as well [1]. However each organization develops its own unique way of collecting and storing the data, locking the information in isolated silos that cannot interoperate with the tools of other organizations. Furthermore,a lack of common data protocols designed to support data sharing makes data exchange and aggregation difficult to achieve. In addition, the lack of IT expertise, and the considerable amount of time and cost associated with building mobile applications prevents the widespread use of mobile applications aimed at automating real-time notifications, damage assessment and coordination of relief operations.

However, data aggregation is very challenging at this point because each organization develops their own way of collecting and storing the data and there is a lack of common data protocols to support the sharing and sending of the collected data. Linked Data technology can help here and one of the important features in Linked Data is data interoperability. It is very difficult to have everyone agree on the term or schema they use for storing and describing the same data. However, when using Linked Data, apps work as long as they use the same ontology to describe the same data, even though they might use different terms to describe the same data. Chapter 3 talks about Linked Data in detail.

The focus of this research is to enable non-programmers, such as volunteers and first responders, to quickly develop and deploy mobile applications designed to help coordinate and manage humanitarian relief operations. In addition, data collection and aggregation is an important aspect of these operations. I've developed a toolkit that enables easy form creation and data collection. The toolkit is built on the Punya framework [1], a system that enables (non-technical) users to develop applications with advanced mobile features, including push notifications, production and consumption of structured data, various visualizations, and wearable device support. I've based my toolkit on the existing Linked Data architecture of Punya to enable the easy development of Linked Data mobile apps that require detailed forms for extensive data collection.

This thesis introduces a DIY toolkit to allow these non-expert workers to conveniently build Mobile Linked Data applications that can meet the needs of the humanitarian relief community. The thesis also describes the Punya project's ongoing collaboration with the humanitarian relief community.

1.2 Contribution

I worked on the Punya framework project extensively as an Undergraduate Research Assistant. I have contributed to many aspects of the project, including the Google Map component, Cloud Messaging component, Google app scripts server code, etc. The specific contribution of this thesis is the following:

- I conducted a participatory design workshop with the people at the International Committee of Red Cross.
- I demonstrated the importance of having mobile Linked Data applications assisting the aid workers during a crisis management scenario.
- I showed that humanitarian workers can develop and deploy mobile applications using the Punya framework given the proper guidance.
- I presented a methodology for guiding humanitarian workers in building mobile apps tailored to their specific needs.
- I discovered possible improvements for the Punya framework that could improve how aid workers build humanitarian mobile applications.
- I introduced the Linked Data Form Extension, for people to easily build apps with a Linked Data form.
- I put together a Virtuoso Docker container for people to bring up an instance of the Virtuoso Linked Data store within minutes with a few simple commands.

1.3 Outline

In Chapter 2, I present background information on the Punya framework - one of the many visual programming languages used for developing mobile applications. Chapter 3 talks about what Linked Data is and how data interoperability works in Linked Data technology. In Chapter 4, I talk about a participatory design workshop and semi-structured interviews conducted with staff and managers from the International Committee of Red Cross. In Chapter 5, I provide details on how humanitarian workers can develop and deploy mobile applications using the Punya framework when proper guidance is provided. Chapter 6 describes the design and implementation of the Linked Data Form Extension, a new feature in the Punya framework developed to help people build apps using a Linked Data form. Chapter 7 evaluates the Mobile Linked Data App Kit by walking through a mobile app development case study using this kit and a collection of templates. Finally, Chapter 8 concludes the thesis and presents future research directions.

Chapter 2

Background

This thesis aims at exploring the the potential for Linked Data mobile apps for the humanitarian relief community, in particular helping relief workers with no technical abilities to be able to build and deploy mobile Linked Data apps. One might think that having relief workers with no technical background develop mobile applications is farfetched due to the time required to learn and be proficient in programming. But nowadays there are many visual programming languages that help novices learn programming quickly. Novice programmers can manipulate graphical blocks and create applications traditionally implemented with text based programming. Such graphical approaches ease many barriers to learn programming and excel as a coder.

My thesis is based on the assumption that relief workers are the *lead users* [22] in the disaster management field. Relief workers are the lead users in the disaster management domain and are the ones who should create the apps since they are the ones who are aware of and understand the problems arising during a disaster.

Lead users expect to benefit significantly by finding a solution to their needs. As a result, they often develop new products or services themselves because they can't or don't want to wait for them to become available commercially. For example, 3M picked up the idea of lead users and assembled a team of lead users which included a veterinarian surgeon, a makeup artist, and doctors for new products. In the end, they developed a breakthrough surgical drape product [22].

In order to understand the work presented in this thesis, it is important to first gain a basic picture of the Punya framework and how it enables people to become mobile app creators. Then, the chapter transitions to a discussion of Punya's features especially designed for humanitarian relief. We will provide an overview of a mobile application built using the framework. The chapter ends by reviewing the motivation behind the Punya framework compared to two other mobile application development platforms for crisis response and management.

2.1 The Motivation behind the Punya framework

Mobile phones are increasingly being used in disaster management as the devices become increasingly equipped with sophisticated sensors. These sensors enable real-time monitoring of the environment and individuals[16]. Mobile applications and the associated services help relief workers perform necessary tasks such as gather and store data regarding real-time issues within a disaster, monitoring population movements to better assess real-time distribution channels, and further complement on-the-ground communication infrastructure.

Given these potential benefits, humanitarian organizations such as the Red Cross and the United Nations have sought to develop specialized mobile applications to be used during disasters. However, there are several challenges that aid agencies face in the development of mobile applications for in-the-wild use. Aid organizations often don't have in-house software development teams. The bulk of mobile application development is often outsourced to third-party vendors [6] with a lengthy software development process that might not fulfill the needs of the critical on-the-ground response quickly enough. These gaps often lead to the development of expensive mobile applications that may not be useful in the actual disaster response situations [6].

Given these shortcomings, an agile disaster management platform that enables humanitarian workers to easily and independently develop, modify and deploy humanitarian-focused mobile applications in real time would be a more ideal solution [6]. This vision motivated the creation of the Punya framework [8].

2.2 Introduction of the Punya framework

The Punya framework [13] is powered by MIT App Inventor technology [12]. App Inventor is a free, block-based, web application that allows people to drag-and-drop visual objects to create Android mobile apps. App Inventor is an open-source project with more than 4 million registered users. They have created over 12 million applications ranging from tracking rainfall in Haiti to guiding blind or visually-impaired students around their schools to teaching US Marines how to destroy improvised explosive devices (IEDs).

2.3 Overview of App Inventor

Punya was built utilizing the technology from App Inventor. App Inventor has 3 major components: the designer, the Blocks Editor, and the Compiler

2.3.1 Designer

The Designer is a mini-workspace for creating a mobile application's graphical interface. There is a *Component Palette* on the left side of the Designer that contains a list of available components to be used to create the mobile app, for example Button, TextBox, Label, as shown in Figure 2-1. In the designer, when one of these components is dragged from the Palette onto the Viewer, it appears listed in the *Components panel*, as shown in Figure 2-2. The *Properties panel* lists editable options used to change the way components look and are displayed. Properties are located at the right side of the Designer as shown in Figure 2-2. People can create multiple screens in the Designer and each screen is simulated to resemble a mobile application interface.

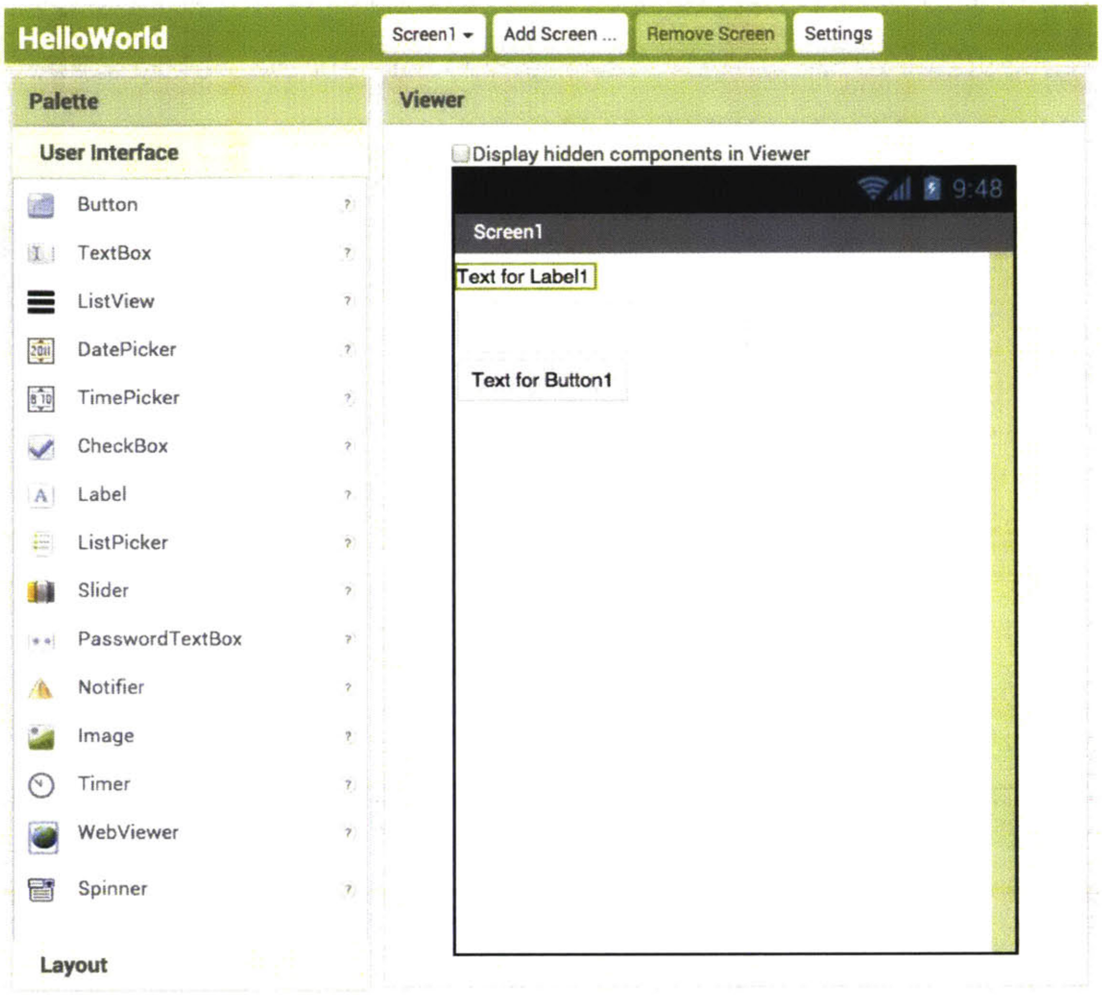
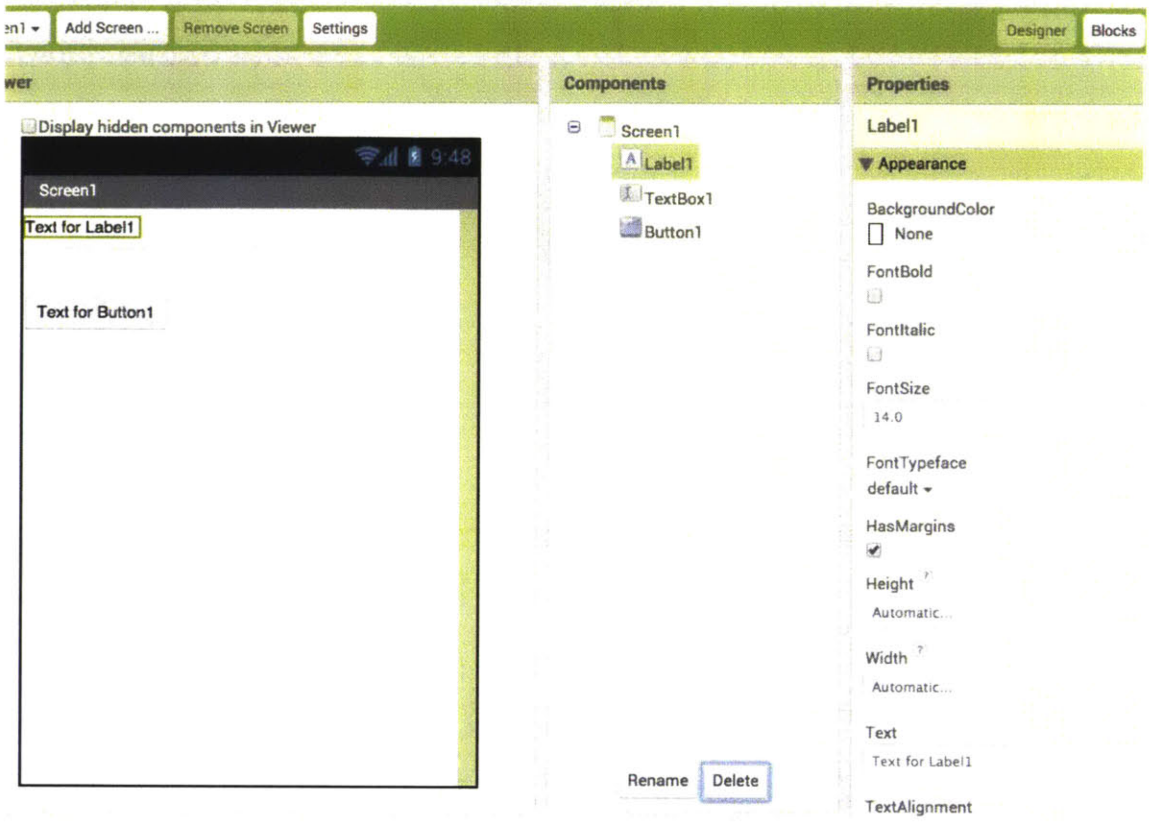


Figure 2-1: Designer interface of Punya framework (left portion) displaying the Palette of components and the Viewer in Screen 1



Label1

Appearance

BackgroundColor
None

FontBold

FontItalic

FontSize
14.0

FontTypeface
default

HasMargins
✓

Height
Automatic...

Width
Automatic...

Text
Text for Label1

TextAlignment

Figure 2-2: Designer interface of Punya framework (right portion) displaying the Viewer, the Components panel and the Properties panel

2.3.2 Blocks Editor

People can program the mobile application logic using the Blocks Editor. On the left side of the interface, there is a *Blocks Palette* that lists all of the available graphical blocks. This includes the core system built-in blocks, such as Text operators, Lists methods, and Color selection, and the component-specific blocks that display components previously selected by the user in Designer, as shown in Figure 2-3. For example, if a user selects a Button component in the Designer by clicking on the Button component, the Blocks view displays a list of methods corresponding to that Button. An example of a relevant method is: “when.Button1.Click”. In the

Blocks Editor, people can freely drag-and-drop blocks from the palette onto the screen to create behavior by defining a sequence of actions. A sample sequence is shown in Figure 2-4.

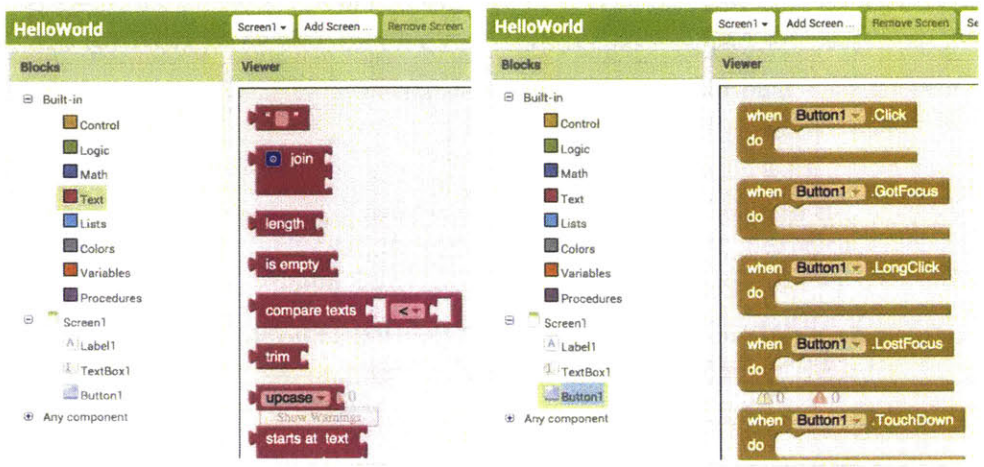


Figure 2-3: Blocks interface with both built-in blocks (shown in the top of palette) and the Button-specific blocks (shown in the bottom of palette)

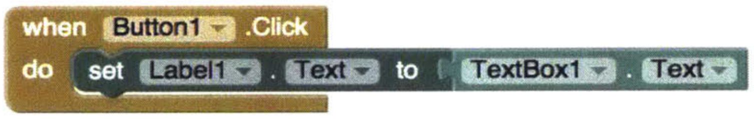


Figure 2-4: Sequence of blocks

2.3.3 Compiler

The Compiler is a service responsible for reading and processing the contents in the Designer window and in Blocks Editor and turning them into an Android Application Package (APK) which is required for the installation in a mobile device. Compiler invocation is listed under the “Build” menu on the top toolbar as shown in Figure 2-5.

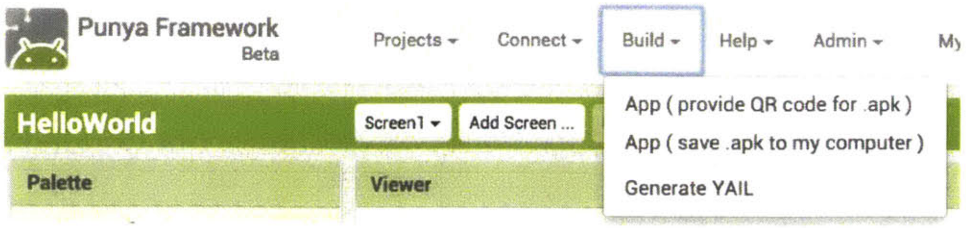


Figure 2-5: The "Build" menu on Punya Framework Toolbar

2.4 Punya framework Highlighted Features

Punya is implemented as an extension of MIT App Inventor. It adds several features designed to support the development of mobile apps specifically for humanitarian relief efforts. The overall architecture for the Punya framework is shown in Figure 2-6. In particular, the five highlighted components in Punya that the team has developed are a better data collection and querying *structured data*, *crowdsourcing*, *sensors*, *wearables*, and *streaming data*.

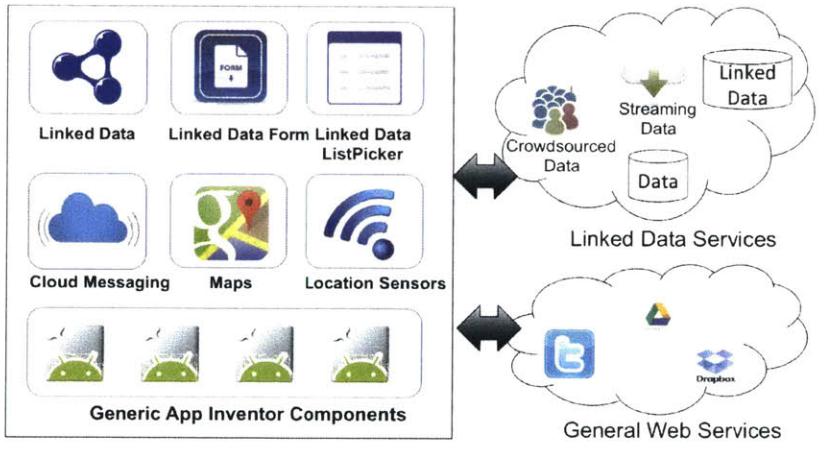


Figure 2-6: App-Building Framework Architecture

A Better Data Collection and Querying Structured Data: One major challenge in crisis situations is how to effectively integrate and use data generated by multiple parties, including the public, relief organizations, and government agencies. The Punya team addressed this challenge by using Linked Data principles to enable the reuse, extension and integration of heterogeneous structured data from different distributed sources. Linked Data is a set of design principles proposed by the World Wide Web Consortium (W3C) [10]. It uses Web technologies, such as the Universal Resource Identifier (URI) to support the distributed development of structured information so it can be easily and automatically combined. Consuming and generating Linked Data is difficult, especially on mobile devices [18]. Another challenge Punya addressed was reducing the barrier to the adoption of Linked Data technologies on smartphones.

Crowdsourcing: The Punya framework's Form elements enable crowdsourcing by supporting the generation of different kinds of forms including those capable of collecting text, media, and various sensors such as GPS location. Its Cloud Messaging Component makes it easy to develop apps that receive notifications in real-time about new data being collected from the public. This allows app users who have subscribed to a specific topic, for example "Fallen Trees", to continuously receive push notifications of reports submitted by other users related to this disaster information.

Sensors: The framework provides components to access 15 kinds of sensor information on a mobile phone through easy-to-use visual blocks. Apps developed in the framework can use these sensors to detect information about what is happening in the background. For example, an app could detect if a user is walking using the built in activity probe sensor and location probe sensor; or detect the ambient conditions using the light sensor. Here is the list of the sensors and their short descriptions:

- *Activity Probe*: records how active a person is
- *Battery*: provides info about the current status of the battery in the device
- *Call Log History:* returns info of recent calls which were made by the device
- *Cell Tower Probe*: detects info of the cell tower that the device is connected to
- *Contact Info*: returns info of the contacts in the device
- *Light*: returns info of the current illuminances (lux) in the device

- *Location Probe*: records GPS locations
- *Pedometer*: counts each step a person takes
- *Proximity:* detects whether or not the phone is close to the ear
- *Running Applications*: tells you what applications are currently running
- *Screen Status*: returns the current screen status in the device
- *SMS History*: messages sent and received by the device
- *Social Proximity*: detects whether or not a Bluetooth enabled device is close by
- *Telephony Info*: gives info about the telephone services on the device, such as cellular strength, service provider, etc.
- *Wi-Fi*: a component that detects information of the nearby Wireless Access Point

Wearables support: The framework has built-in support for smart watches and allows notifications from the app to be displayed directly on the watch.

Streaming: The framework has a streaming component that can receive information from a streaming data source in near real-time. It also has the capability to filter data which delivers the most relevant data to the app user.

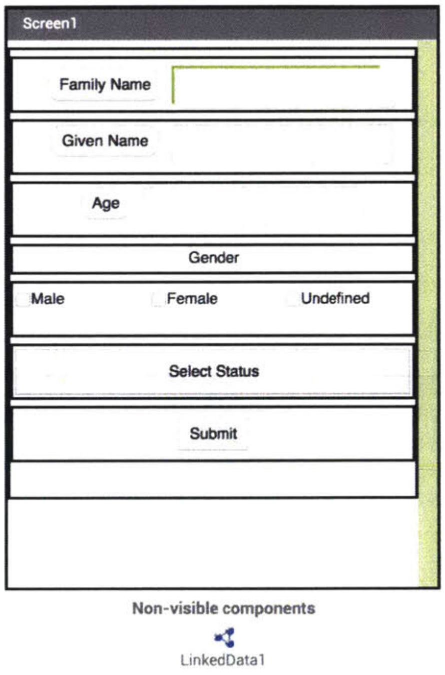
2.5 Linked Data Components in the Punya framework

Consuming and generating Linked Data is difficult, as pointed out in Scharffe et al. [19], especially on mobile devices. Punya therefore includes features aimed at reducing barriers to publishing and consuming Linked Data on smartphones.

2.5.1 Publishing Linked Data

The Linked Data Form component is a layout component that can be used by developers to identify a collection of fields that should be applied to a particular ontological concept [19].

This component was built as part of Punya to enable forms to be created and thus makes data collection easier. A form consists of numerous fields used to enter data. Each field in the form holds a field label so that any user who views the form gets an idea of its contents. Each field in the Linked Data form contains the property URL as the identifier. For example, a developer can have three text fields for the family name, given name, and age using the Linked Data Form component as shown in Figure 2-7. Figure 2-8 explains the steps using blocks to take a Linked Data Form component, convert it into triples, and publish to the web.



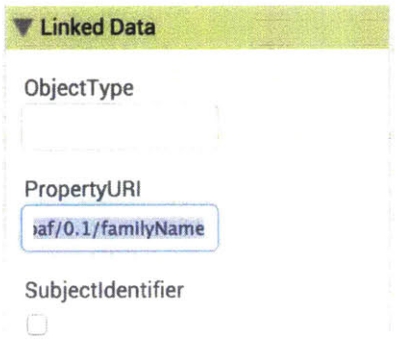


Figure 2-7: A Linked Data form (left); the family name field has the PropertyURI *"http://xmlns.com/foaf/0.1/familyName" (right)*

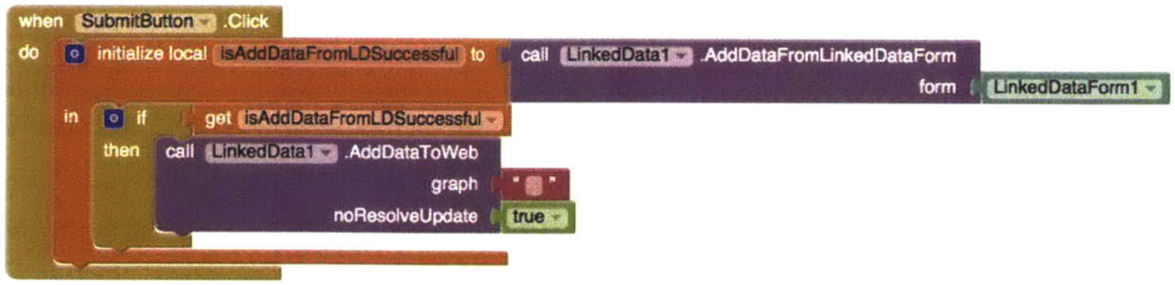

when SubmitButton .Click
do initialize local isAddDataFromLDSuccessful to call LinkedData1 .AddDataFromLinkedDataForm
form LinkedDataForm1
in if get isAddDataFromLDSuccessful
then call LinkedData1 .AddDataToWeb
graph
noResolveUpdate true

Figure 2-8: A collection of blocks which take a Linked Data Form component, convert it into triples, and publish to the web.

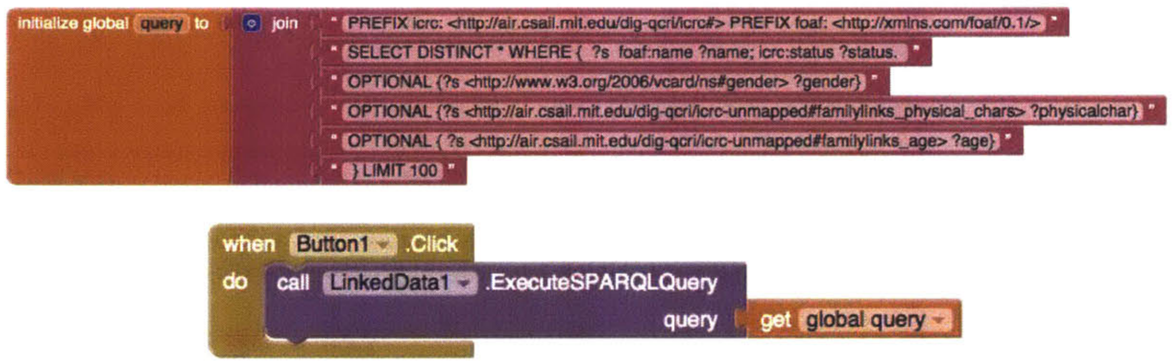
Because we focus on data interoperability, the form elements automatically and transparently create and store structured data. The framework also includes a Linked Data Query Component that is able to execute SPARQL Protocol and RDF Query Language (SPARQL) queries against remote Linked Data stores [20]. RDF stands for Resource Description Framework and it is a metadata data model created by W3C. Data collected using RDF is structured data. All of Punya's Linked Data components are compliant with existing W3C standards and are able to take advantage of community standards such as Management of a Crisis (MOAC) and the Humanitarian eXchange Language (HXL) vocabularies which will be explained in Chapter 3.

2.5.2 Consuming Linked Data

The Linked Data Component converts Linked Data forms into RDF graphs, executes and processes results of SPARQL queries, and saves and loads the content of ontologies [19]. For example, an app developer can execute a SPARQL query to fetch the first 100 entries about people in the triplestore and return the results as text. Figure 2-9 shows the query and Figure 2-10 shows the blocks needed to execute this SPARQL query.

```
PREFIX icrc: <http://air.csail.mit.edu/dig-qcri/icrc#> PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT DISTINCT * WHERE {
        ?s foaf:name ?name; icrc:status ?status.
        OPTIONAL {?s <http://www.w3.org/2006/vcard/ns#gender> ?gender}
        OPTIONAL {?s <http://air.csail.mit.edu/dig-qcri/icrc-unmapped#familylinks_physical_chars>
        ?physicalchar}
        OPTIONAL {?s <http://air.csail.mit.edu/dig-qcri/icrc-unmapped#familylinks_age> ?age}
} LIMIT 100
```

Figure 2-9: SPARQL query in text to fetch the first 100 entries about people in the triplestore



```
initialize global query to  join  " PREFIX icrc: <http://air.csail.mit.edu/dig-qcri/icrc#> PREFIX foaf: <http://xmlns.com/foaf/0.1/> "
                                  " SELECT DISTINCT * WHERE {  ?s  foaf:name ?name; icrc:status ?status. "
                                  " OPTIONAL {?s <http://www.w3.org/2006/vcard/ns#gender> ?gender} "
                                  " OPTIONAL {?s <http://air.csail.mit.edu/dig-qcri/icrc-unmapped#familylinks_physical_chars> ?physicalchar} "
                                  " OPTIONAL { ?s <http://air.csail.mit.edu/dig-qcri/icrc-unmapped#familylinks_age> ?age} "
                                  " } LIMIT 100 "

when Button1 .Click
do  call LinkedData1 .ExecuteSPARQLQuery
                                 query  get global query
```

Figure 2-10: The blocks construct the SPARQL query and execute this SPARQL query

It's too hard for beginning users to create SPARQL queries at this point. The toolkit presents in this thesis simplifies the SPARQL queries creation process. Section 6.3.1 talks about that in detail.

2.6 Related Works

There are numerous mobile application development platforms being created. Some are specifically designed for crisis management. Few of them are designed for data collection using forms.

One system which focuses on data collection is Kobotoolbox [4]. Kobotookbox was designed by researchers at the Harvard Humanitarian Initiative to collect data from survey questions. It utilizes web services so people can use their own interface to build survey forms by drag-and-drop actions, or reuse existing forms from the community and then load the forms onto mobile phones. Users of the Kobotookbox have the option to send the data to the Harvard Kobotookbox data server, or to a third party Kobotookbox compatible data server. Sending the data to the Harvard Kobotookbox server is the default option. Once the server has the most updated data, users can start to analyze the data on the Harvard Kobotookbox web server. Kobotoolbox works across multiple platforms such as mobile and desktop computing to enable quick and seamless data collection in the field during an environmental disaster. Workers on the ground can use Kobotoolbox to easily collect survey information as well as other data, such as GPS location, images, etc. However, Kobotoolbox is not designed for building general disaster mobile applications since it doesn't provide support for features beyond survey forms. A messaging feature, for instance, is a typical and useful feature needed in many disaster situations. In addition, data integration and interoperability is not addressed in this platform as well.

AppMakr.com is another Do-It-Yourself platform for building native mobile apps for iPhone & Android OS [15]. The platform is mainly driven by the app templates and users modify existing app templates to create new apps. It comes with a What-You-See-Is-What-You-Get (WYSIWYG) editor and requires no prior coding knowledge for building mobile apps. It focuses on creating content-based apps. However, AppMakr is not ideal for building general disaster mobile apps since it is template driven and provides little in the way of customization. In addition, data collection is almost impossible in this platform because it requires access to the sensors and AppMakr doesn't provide such access.

At this point, I haven't encountered any general framework or platform which provides easy mobile app creation for people with no programming experience, and which also supports data collection using forms and addresses the data integration and interoperability capacity. This motivated me to create a DIY kit for developing mobile Linked Data apps for humanitarian assistance. This kit should specialize in data collection using forms and it should come with easy data integration and interoperability.

Chapter 3

Linked Data

One major challenge many organizations face nowadays is how to effectively integrate data generated by multiple parties. This challenge is even more difficult when there are non-technical people involved in the process. Data integration requires a lot of effort, especially when the database schemas are different even though they describe the same things. For example, one of the challenges data administrators face is that it is hard to built applications to be data integration-friendly because people keep altering the applications. Requirements in the applications always change along the way and it is very hard to get everyone to agree on a plan or data schema standards and make no changes for a period of time. We use Linked Data principles to address this challenge. Linked Data technology enables the reuse, extension and integration of heterogeneous structured data as noted in Chapter 2.

Previous researchers have shown the uses of Linked Data within the mobile device environment are useful and challenging at the same time. David et al. [7] propose a general framework in order to introduce Linked Data sources as device content in the Android platform. This general framework enables applications to deliver their data in RDF and to exploit this information in a uniform way. The framework David presented also allows people to connect this information to the semantic web and the web of data through reference from and to device information [7]. D'Aquin et al. demonstrate a mobile application called wayOU (where are you at the Open University) which relies on the data published by The Open University (under data.open.ac.uk) to provide social, location-based services to its students and members of staff [11]. The app itself consumes Linked Data produced from various repositories in the university. It creates new connections between people, places and other types of resources as Linked Data.

Razzak et al. presents the design and architecture of an integrated environment that provides location independent and mobile access to Intelligent Domotic Environments [17]. This architecture provides users with the ability to acquire information about devices in the Linked Data formats.

The approaches listed above are domain specific, which implies that they are very difficult, or impossible extend to be used with different data sources, or even to be used within a general framework. However, the Linked data components in the Punya framework enables users to reuse available Linked Data sources. It also allows users to tailor and extend apps to a variety of scenarios, not only limited to humanitarian and/or relief works. There are two parts, the mobile Linked Data app and the Linked Data store when we talk about Linked Data here. The Punya framework currently provides support for building the mobile app, but not the Linked Data store. Users of the framework need to host their own versions of the Linked Data store.

3.1 Linked Data Principles

Tim Berners-Lee is the father of Linked Data and he outlined four principles of Linked Data in his Design Issues [2]:

- Use URIs as names for things
- Use HTTP URIs so that people can look up those names.
- When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL)
- Include links to other URIs. so that they can discover more things.

3.2 Vocabularies for Linked Data

Activities such as setting up homeless shelters, reporting damages, overseeing volunteers, and coordinating and managing relief operations in disaster management often generate a lot of data. These data could be essential in decision making processes during disasters if they integrated and used by different data sources. Previous research [3] has demonstrated that using ontologies and Semantic Web technologies can be beneficial in developing applications for disaster management scenarios. People in the disaster management research area use two data standards HXL and MOAC. The Punya framework has used these vocabularies in the Linked Data implementation.

3.2.1 Humanitarian eXchange Language Situation and Response Standard (HXL)

HXL[5] was developed through an initiative by the United Nations High Commissioner for Refugees (UNHCR) after inspecting a broad range of systems in use by different humanitarian organizations. The schema is available in five main sections: geolocation, humanitarian profile, metadata, response, and situation. The geolocation section is for information about the locations affected. The humanitarian profile section includes the information on the populations affected. The metadata section asks who is collecting the data, date and time of when the data was collected, etc. The response part contains the information on the organization responding. Lastly, the situation section is about the information related to the emergency.

3.2.2 Management Of A Crisis Vocabulary (MOAC)

MOAC [9] was the very first Linked Data vocabularies developed to to help non-experts generating Linked Data formats for disaster management. It was created after the earthquake in Haiti in 2010. MOAC is defined as a set of classes and properties and it contains the suitable vocabularies for disaster management. MOAC is used to to link crisis information from traditional humanitarian agencies, volunteer and technical committees, and disaster affected communities.

3.3 Data Interoperability

One of the important features in the Linked Data is data interoperability. It is good to have everyone agree on the term they use for describing the same thing. But this is very challenging and only a few organizations can or have agreed to do that. There are many obstacles preventing them from doing so and constant changes in application requirements, such as data schema is one of the big issues.

However, apps built when using the same ontology for describing data can query datasets in and from different RDF stores. This works as long as they use the same ontology to describe the same data, even through when they might use different terms to describe the same data. For example, suppose organization A refers to people killed in an disaster as "dead" and organization B refers to these as "deceased". Traditionally, there is no way to merge the data from those two sources unless there is a table for mapping "dead" with "deceased". Linked Data can help in this situation if the data collection is setup correctly at the beginning. First of all, before either organization A or organization B start to collect data, they need to use the identifier for describing the "dead" or "deceased". According to the Management Of A Crisis Vocabulary, the closest HTTP URI is:

http://observedchange.com/moac/ns/#Deaths

As long as both organization A and organization B use this HTTP URI, they can interoperate or integrate these two datasets by issuing a federated query. Federated query execute SPARQL queries distributed over different endpoints. This works despite there are two different terms describe the same thing.

The Punya framework helps organizations to achieve data interoperability in two aspects. First, it makes makes it easy to develop mobile Linked Data apps that collect data in a structured data format. Second, it provides standardized ontologies during structured data creation. For example, the Punya framework includes the Humanitarian eXchange Language Situation and Response Standard and Management Of A Crisis Vocabulary in the ontology collection of the disaster category. The framework will select the terms or ontologies from these two sets of vocabularies first.

3.4 Mobile Linked Data Applications

We describe two applications that can potentially be used by relief workers during a crisis situation: (1) *WeReport* App which could be used to report scenes and information related to the disaster. This app will have the ability to be integrated with different public datasets using linked data, and (2) *Donate-N-Request* App which will be used to match requests and donations during a disaster scenario. The toolkit which I present in this thesis can build these kinds of mobile apps. In addition, it provides more functionalities beyond what Punya can already support at this point. Chapter 7 of this thesis talks about the additional features in detail.

3.4.1 WeReport App

The *WeReport* app as shown in Figure 3-1 is an example application that was developed using the Punya platform, *WeReport* is an incident crowdsourcing application that allows individuals to easily report disaster incidents through texts, pictures and GPS locations using an Android mobile device. Individuals can also easily follow (by subscribing) various kinds of ongoing incidents occurring in their vicinity. The *WeReport* application demonstrates how information from smartphone sensors and other sources can be easily integrated in a mobile application. The ease of the use-modify-create mechanism enables other individuals to develop additional capabilities on top of the *WeReport* application. The *WeReport* application can be extended to report power outages, road accidents, fire assessments, etc. The ability to quickly develop and deploy an application like *WeReport* presents a unique opportunity for crisis managers and other volunteers to more easily leverage mobile technology during disaster scenarios.

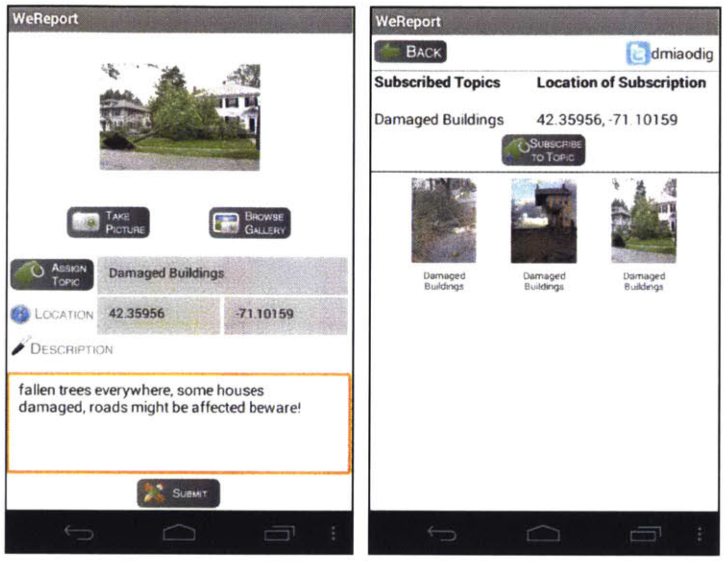


Figure 3-1: The user interface (UI) for the reporting and subscription in *WeReport* allows people to submit disaster reports to the cloud. People can also subscribe to reports close to a certain location or on a topic such as "Fallen Trees" and subscribers receive real-time results of these topics

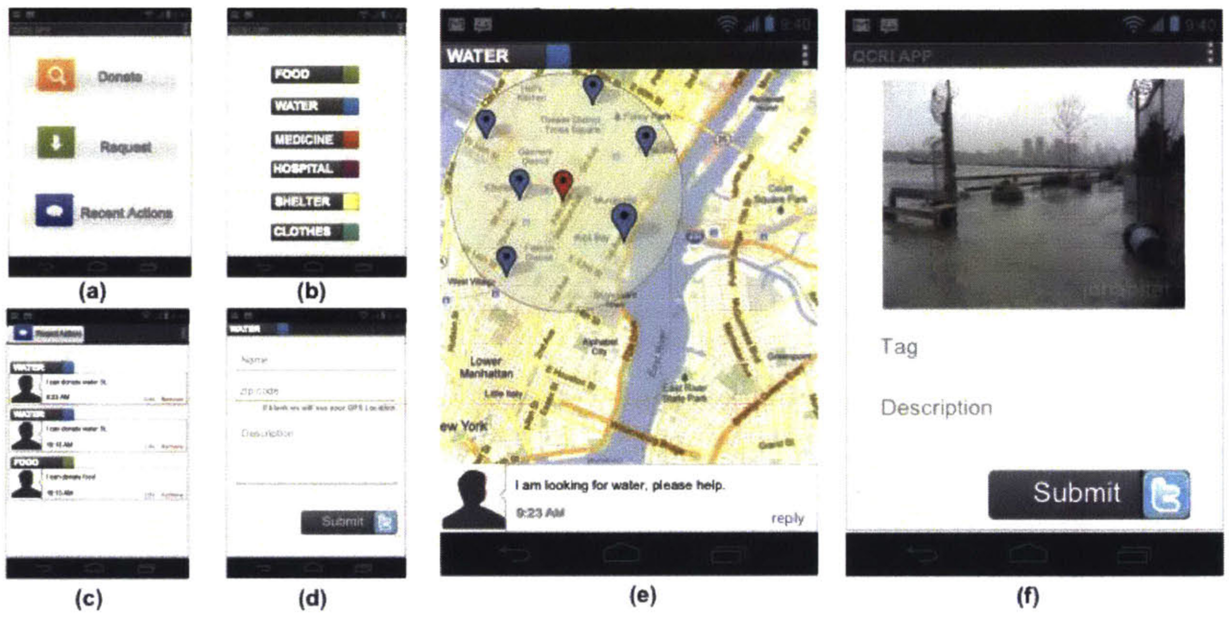
3.4.2 Donate-N-Request App

The *Donate-N-Request* app is inspired by MatchApp[1], where the demand for resources is matched with the supply. Consider this resource match-making scenario: Alice lives in New York City. After Hurricane Sandy hits, Alice wants to help people in need. Bob has been

[1] http://irevolution.net/2013/02/27/matchapp-disaster-response-app/

affected by the crisis; Both Bob, the requestor, and Alice, the donor, can place the donation and request for items within our app as shown in Figure 3-2a & 3-2b [19].

As shown in Figure 3-2c, users of the app can also view, edit, or remove recent list of their donated or requested activities. For example, Alice can donate water to people who are within a three-mile radius of her location. The map in Figure 3-2e pinpoints the requests. She selects Bob from the list of people in need as Bob had previously requested using our app (Figure 3-2d). Once the selection is done, they can communicate directly by text message with each other.



I am looking for water, please help.
9:23 AM

Figure 3-2: Two Linked Data enabled mobile apps for disaster response situations

Chapter 4

Participatory Design Workshop

There are several barriers that impede the development and deployment of mobile apps in humanitarian organizations. Firstly, it is very costly and time-consuming to build and deploy mobile applications(~$200,000 - $300,000). It often takes one to two years to complete the entire development and deployment cycle. In addition, changes are often required halfway through or near the end of the project. Due to these moving goalposts, the final mobile app might not achieve full functionality before the project deadline. These are some of the issues highlighted in interviews during the workshop with the International Committee of Red Cross (ICRC) and by other aid organizations.

In order to understand common and relevant issues, I conducted a participatory design workshop with staff members and managers from ICRC. This workshop was also conducted in order to gather possible ways to address the issues mentioned above. The two-day workshop took place in the ICRC office in Geneva, Switzerland from October 16th to October 17th in 2014 and participants represented a range of teams within the ICRC. Overall, the response was enthusiastic because the Punya framework allowed them to quickly prototype mobile apps based on their ideas and needs.

The goals for the workshop were the following:

(1) participatory design workshop for designing mobile apps for first responders;

(2) train and familiarize the ICRC staffs with the Punya framework; and

(3) evaluate the Punya framework.

Participants in this study built and deployed humanitarian-focused apps to accomplish pre-selected tasks. There were six tasks, for example, one of the tasks was to build a mobile app to query find the names of the all the dead from a known database; another task was to dead body to the existing database. A comprehensive list of the apps created as part of this study can be found in section 4.2. I conducted interviews after they done with the pre-selected tasks. During interview sessions, participants gave insight into the current practice of developing and deploying mobile apps within their organization and the problems associated with their current process. Later, participants shared their experience using the Punya framework, potential mobile apps for their projects, and suggested improvements for the framework. It was a two-day workshop and it run from 9:00am to 5:00pm with an hour and half break in between. The tutorials was about 6 hours, discussion was about 3 hours, and interviews was 5 hours.

4.1 Workshop Methodology

Punya framework training and the interviews were the major parts of the workshop. The following is an agenda for the participatory design workshop:

(1) Introductions from the MIT Punya team and ICRC teams
(2) Participants get training on six tutorials with the Punya framework
(3) Participants come up with one mobile application they want to build
(4) Participants finalize the mobile application design by gathering the requirements and features they want to have the one mobile app
(5) Instructors of the workshop build the working prototype of the mobile application overnight
(6) Instructors gather feedback of this mobile application from the Participants
(7) Participants evaluate the Punya framework. We gather more functional requirements for the an app-building framework and to look for more opportunities for mobile apps from other team within ICRC

This agenda is for the participatory design workshop for first responders. Step 1 to step 3 cover the training to the ICRC staffs with the Punya framework. Step 4 to step 7 cover the evaluation of the Punya framework.

4.2 Day 1 Workshop

The first day of the workshop, we included a training session with five ICRC staff and three managers. Three out of eight participants had basic to intermediate programming experience and the remaining five participants didn't have any. Participants underwent comprehensive training for the Punya framework. The study began with with a presentation exploring the functionalities of the framework, the technologies that power it, and a demonstration of some apps developed using Punya. After the presentation, the participants worked through six tutorials ranging from basic to advanced. Below is the list of the six tutorials.

1. Simple listing of people from the given people database
    a. Define and execute a SPARQL query to look for all the entries in the people database.
    b. The people database contains entries which have people listed as alive, missing, injured, and dead.
    c. Output the result as raw text
2. Simple listing of people from given people database, in well-organized format
    a. Each entry in the people database has the name, age, gender, physical characters fields, etc.
    b. Output the each entry with just name, age, gender, and the physical characters fields.
3. Query people by status from the people database
    a. Modify the task 2 by adding a people status option in the SPARQL query.
    b. There are four options for the people status: alive, missing, injured, and dead.
4. Query people by geo-location from the people database

  a. Modify the task 2 by adding a search radius option in the SPARQL query.
  b. Users of the application need to supply the search radius and only the entries which meet the search conditions can be displayed.

5. Report people to the database
   a. Build a Linked Data form which contains the family name, given name, age, gender, and status fields.
   b. User can submit this form to the people database.
6. Report building to the database
   a. Modify task 5 to build Linked Data form which contains the building name, address, GPS location, damage level, and description.
   b. User can submit this form to the damaged building database.

The Punya team designed the tutorials to relate to one of the ICRC's projects, Restoring Family Links. The Restoring Family Links project is about reconnecting families separated either domestically, or internationally. For example, during a crisis first responders on-site report a missing person's information while others report a found person's information, and then aid organizations attempt to match the information. Typically, all reports are hand-typed and greatly prone to human error, such as typos and missing information. Such errors increase the complexity involved in reconnecting family members. A digital approach for reporting with minimal human error is the desired solution. Therefore, the tutorials asked the participants to develop and modify mobile apps to query for missing people in the database and to report missing people when they are found.

The participants were surprised at how easily they could build a mobile app using the Punya framework. They were new to mobile app development and they finished all six tutorials within six hours. One participant said -

"I wouldn't imagine I could write a mobile app. I can finish the mobile apps I want within hours. That is awesome!"

Despite the overwhelming positive experience, two participants pointed out that it could be difficult to locate specific logic/programming blocks in a project which has many blocks.

We moved on to the next agenda in the list as shown in section 4.1 once we finished the tutorials. Participants were put into groups and were asked to come up with one mobile application they want to build. The participants identified the mobile apps they want to build and consolidated all of the features deemed necessary for one mobile app to have ideal functionality. The project includes looking for family members, restoring contact, reuniting families, and seeking to clarify the fate of those who remain missing during natural disasters or conflicts. The participants listed the following functional requirements for the app:

- Broadcast the mobile app user's status (alive, in danger, etc.) to a central server and to predefined emergency contacts
- Submit a found or missing person report that includes the picture, name, and geo-location of the person to a central server
- Subscribe to a specific report type (found/missing) and to reports which can be filtered based on geo-location
- Receive real-time updates based on the subscription in the previous step

Day one of the workshop ended with the goal of developing a prototype application for the Restoring Family Links project. I had the specification and design. And the workshop staffs and with the one team member from MIT did the implementation that evening remotely, as preparation for day two.

4.3 Day 2 Follow-up

Day two of the study started with a demonstration of a working mobile version, as shown in Figure 4-1 and Figure 4-2. This app was developed in collaboration with another Punya team member back in MIT remotely. The entire development and deployment process took about four hours. Each of the features required for the application was demonstrated to the participants. Participants were surprised and satisfied with the application.

As shown in Figure 4-1, users of the app need to first register for the app with their personal information. Users need to fill in an emergency contact, which will be used at a later time. Once people complete this step, there are two modes for people to select - safe or emergency mode. In the safe mode, people can send an "I'm alive" message to their emergency contact by pressing the center green circular button. Likewise, an emergency message will be sent by pressing the red button in the emergency mode.

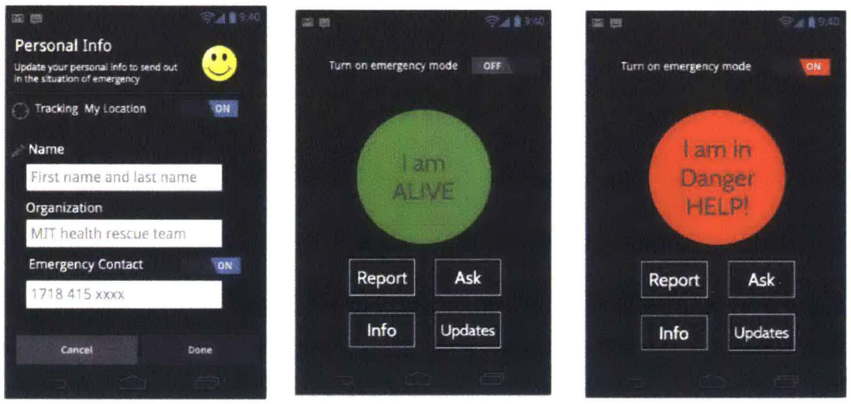
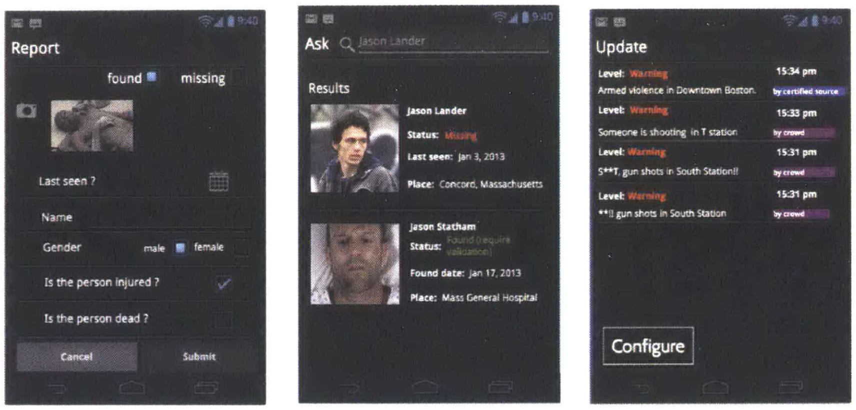

Personal Info
Update your personal info to send out in the situation of emergency
Tracking My Location ON
Name: First name and last name
Organization: MIT health rescue team
Emergency Contact ON: 1718 415 xxxx
Turn on emergency mode OFF
Turn on emergency mode ON

Figure 4-1: Restoring Family App UI
left: personal information
center: safe mode
right: emergency mode



Jason Lander
Status: Missing
Last seen: Jan 3, 2013
Place: Concord, Massachusetts

Jason Statham
Status: Found (require validation)
Found date: Jan 17, 2013
Place: Mass General Hospital

Level: Warning	15:34 pm
Armed violence in Downtown Boston.	by certified source
Level: Warning	15:33 pm
Someone is shooting in T station	by crowd
Level: Warning	15:31 pm
S**T, gun shots in South Station!!	by crowd
Level: Warning	15:31 pm
**!! gun shots in South Station	by crowd

Figure 4-2: Restoring Family App UI

left: report found/missing people

center: subscribe to found/missing people report

right: real-time updates

Reporting and receiving real-time reports are another critical feature in the app. Users of this mobile app can report found or missing people and subscribe to report about found or missing people as seen in Figure 4-2. The system will send out the subscription the phone automatically and the phone will receive the subscription almost instantly.

During the session, one participant enquired about the topic of data interoperability. He presented an example related to ICRC: ICRC has three subsidiary organizations, American Red Cross, Canadian Red Cross, and Mexican Red Cross. Each of these use a similar version of the demo app. Each Red Cross modified the demo app by adding or removing fields. In addition to the field changes, each version of the app is internationalized to the local language. This makes data integration harder since the original data schema is changed. This is the typical situation multiple organizations face nowaday. Most of time, people need to manually rebuild the whole databases before they can do any integration.

However, data integration can be easy when using Linked Data technologies. When using the Punya framework, users of the framework can use the Linked Data Form component to turn user input into RDF data,described in common ontologies such as Humanitarian eXchange

Language and Management Of A Crisis which are specified by the app developers. Two apps that use the same ontology for describing its data can query datasets in different RDF stores, even though they have different data schemas.

By using the app-building framework, participants were able to create the fully functional apps (the six tutorials). One of the participants brought up a mock-up mobile app , which he was working on for the past year and had very similar functionalities to the demo app. He explained that the mock-up app was needed by the first responder community, but it was never developed internally due to many problems, such as financial, time, etc. We turned the mock-up idea to a fully functional mobile app. As remarked by one of the participants:

"This tool can really save our time, money, and even lives."

Several participants really appreciate the idea of easy customizations for apps. The scenario these participants described is that people in the humanitarian organization can reuse the this app as template. This template app can meet the their needs with little customizations. Our app-building framework allows its users to upload an existing app, modify the app, and create quickly a new app. For example, one organization might need to localize the language in this template app and users of the app-building framework just need to replace the language strings easily; other might want to ask for more detailed information in the report by adding more fields in the report form using the framework as well. This use-modify-create mechanism is greatly favored by our ICRC participants.

We had a discussion on how to improve the demo app previously built. Below, there is a list of three suggestions which participants agreed with and reported as part of the study:

- Add the ability to notify the ICRC workers in a predefined area
- Provide information about nearby ICRC offices
- Update volunteers or first responders based on their geo-location

4.4 Interviews

The first interview was conducted with the 6 managers. These managers worked in the Digital Communications, Data Protection Services, Innovation Unit, Climate Center, Population Protection Unit, and Restoring Family Links departments within the ICRC. The goal was to identify possible improvements for the Punya app-building framework. The interview began with an introduction of the framework, and a quick demonstration of the Restoring Family Links mobile app that was built during the workshop. We then discussed the additional requirements needed in the the framework so that it could be used by the ICRC (or other ICRC-like humanitarian organizations) during real crises.

Three key features were identified: (1) data encryption, (2)protection and offline operation. Data encryption and protection consisted of ensuring that all the stored and transmitted data between mobile apps and third parties would be encrypted. In the event of the lost or stolen devices, only trusted parties have the ability and the credentials to decrypt and view the data. With regard to offline operation, managers emphasized the need for offline storage, offline operation. For instance, a mobile app with the built-in function to submit the data to a remote server should be able to cache the request of transmitted data when no network connection is available and be able to resubmit the request when a network connection is restored.

Data encryption and protection are ICRC's primary concerns. ICRS wants to minimize the risk of an hostile foreign regime to be able to hack into any of the infrastructures and identify individuals to target or attack. Humanitarian-focused mobile apps could also benefit from working without an Internet connection. A classic scenario for example is a first responder wanting to enter data while visiting shelters, and then have this data automatically synced with a server when he returns to the office. The ability for apps to continue to work while "offline" is a critical piece of functionality that the Punya framework must support based on participants' responses. These types of mobile apps can help the organization to increase its impact and effectiveness.

A second interview was conducted with 13 staff members from the Innovation Unit, Weapons and Decontamination Unit, Forensics, Security, and the Red Cross in Africa at the

ICRC office. The goal was to identify additional framework capabilities needed by the ICRC. At the end of the interview, we identified two more use cases from this meeting:

- Dead Bodies Identification Form: an ICRC worker or other authorized person needs to record the discovery of human remains, so that these can be documented and later possibly cross checked for identification.
- Missing Persons Form: an ICRC worker or other authorized person needs to conduct an interview with someone reporting a missing person (friend, relative, etc.), and document the missing person's details.

Throughout the entire workshop, we found that all the participants wanted to develop a set of template mobile apps for crisis response and management. The idea is that this set of template apps could be easily modified and deployed by first responders from the ICRC's subsidiary institutions, such as the American Red Cross, the African Red Cross, etc. The participants agreed that our app-building framework was an ideal solution to meet their mobile development needs. It would help their projects reach more people, thereby greatly increasing their impact. Their conclusion was that aid organizations should utilize our app-building framework for future mobile app development and deployment.

4.5 Summary

Here is the list of the main findings categorized within each goal set at the start of the workshop.

- Participatory design workshop for first responders
    - Mobile apps can aid humanitarian organizations to increase their impact.
    - The current time and financial investment to build and deploy mobile apps is unnecessarily high.
    - The Punya framework can help reduce the cost and shorten the development time.
    - Participants want to have a template app that has reporting and subscribing features.

- Train and familiarize the ICRC staffs with the Punya framework
    - They completed all the tasks correctly and on time.
    - Using the framework, participants were able to create apps based on concepts or ideas they had developed in 2013.
- Evaluate the Punya framework
    - It allowed them to quickly prototype mobile apps based on their ideas and needs.
    - Users found it was difficult to locate specific logic/programming blocks in a project which has many blocks.
    - The lack of the copy and paste function made it harder to create Linked Data forms. They needed to repeatedly drag and drop essentially the same form elements.
    - Data encryption, protection, and offline operation were found to be essential features which should be included in the Punya framework.
    - ICRC wants to build apps which include dead bodies identification forms and missing persons forms.

4.6 Discussion

The core complexity in deploying technologies for crisis management lies in integrating heterogeneous information from various sources. Further, as described in previous sections, agility and flexibility for developing prototypes for rapid testing are key factors to successfully adapt technology in disaster relief scenarios. In this section, I summarize how the app-building framework and its supported technologies address these challenges. Specifically, I highlight the use of the framework to (a) resolve issues in data integration, (b) support agility in app development and collaboration between entities, and (c) allow customization of apps tailored to local needs. The discussion introduces better practices for future mobile app development and deployment processes.

4.6.1 Data Integration

The app-building framework provides solutions for generating and consuming linked data on mobile devices. Currently, data collected by different apps are not interoperable because these apps are created by different organizations using backend databases that store data in different schemas. For example, one app that is created by organization X for collecting refugee information cannot be integrated directly with data for disaster relief held by another organization Y. These two apps and their backend systems were developed in silo and their data hosted in relational databases lack global identifier needed for data integration. This makes it very hard for either organization to accomplish any data integration automatically.

Apps created with the Punya framework can generate "RDF" data that can be easily integrated with other data represented in RDF. For example, a data-reporting app can use the Linked Data Form component to create surveys for collecting user input. The Linked Data Form component turns user input into RDF data described using ontologies such as Humanitarian eXchange Language and Management Of A Crisis. Apps using the same ontology for describing data can query datasets in and from different RDF stores as discussed in section 3.3 of Chapter 3.

4.6.2 Fast Prototyping for App Development

The capability to quickly create an app and modify it rapidly is critical to crisis response management. During the workshop (as described in the previous sections), we identified some of the main barriers to developing mobile apps for crisis response or humanitarian projects. The potential challenges include: (a) requirements gathering (b) validating critical features (c) estimating a budget (d) changing requirements, and (e) rapid testing.

In many cases, a new project in a Non-Governmental Organization or Nonprofit Organization is a direct response to an ongoing crisis. While the needs for crisis management might be obvious (e.g. saving human lives, restoring damages, coordination of crisis resources), integrating them as requirements for a potential app requires collaboration from multiple stakeholders.

The discussions identified critical features that leverage existing resources and address the problems at hand. For example, in the ICRC case, the potential app for the Restoring Family Links project needs to fetch current GPS location of the person reporting the status of a missing or found person. Decision makers for the Restoring Family Links project may want to receive real-time notifications of changes within certain monitored areas(e.g. an increasing number of missing people). Quick prototyping and rapid testing of an app helps to refine the requirements on the fly, which might have not been considered when first designing the app. For example, a new requirement for automatically tagging all reports created by the app with cached locations might emerge after finding it requires a lot of energy and computational time to renew the GPS location.

4.6.3 Customization of Apps

Throughout the workshop, we found that the feature to reuse and modify an existing app is a very much needed feature by our ICRC participants. For example an app built for an international organization like the ICRC should be implemented so that it could be personalized to be used or training by different local branches, hence have the flexibility to be customizable to meet the needs of each local branch. he master app can be an e-book app that instructs volunteers on how to participate in disaster relief efforts. However, there might be emergent needs locally that require quick changes to the app. The app-building framework allows its users to upload an existing app, modify the app, and create a new app anytime they need. This use-modify-create mechanism allows different entities to take apps created by others and improve on or customize it to suit their immediate needs.

Chapter 5

Developing Mobile Applications with ICRC Project Manager

The goal of this chapter is to report on how humanitarian workers can develop and deploy mobile applications using the Punya framework when proper guidance is provided. I have presented a methodology of best practices for mobile application development in section 5.4. It is based on the existing best software development practices. The remaining sections in this chapter talked about the experiment and the related findings from testing this methodology and section 5.11 introduced an improved methodology. The experiment was done with an ICRC non-technical project manager through developing mobile applications using the Punya framework.

This chapter describes a mobile app building experiment conducted with one of the non-technical managers at the International Committee of the Red Cross. The chapter describes how humanitarian workers can better design, prototype, build and deploy mobile app. It is in accordance with the needs of their projects following a mobile app development methodology.

This chapter first summarizes the critical elements of disaster relief efforts from the participatory workshop and interview in Chapter 4. I then describe a hands-on mobile application development experiment, where I guided an ICRC manager in creating an app that draws on the workshop results. Lastly, this chapter discusses the findings from this experience and presents a set of methodologies for guiding humanitarian workers in building mobile apps tailored to their specific needs.

5.1 Experiment Plan

The application development experiment lasted for six weeks, from late January to Mid March of 2014, with one of participants in the design workshop who had no prior programming experience before this experiment. The participant in question will be identified as A in the rest of this chapter. During the workshop, A designed, prototyped and developed the Dead Bodies Identification app and Missing Person app while I was guiding A through the process. A is based in Geneva, Switzerland and I am based in Massachusetts, USA. We connected weekly and each meeting lasted for approximately one hour. At the end of each meeting, I assigned tasks and exercises to the A. We utilized *Google Hangout* to communicated A successfully built two mobile applications at the end of our virtual meetings. The following is an agenda for the experiment:

- Finalize the important scenarios encountered in the disaster relief efforts from Chapter 4
- Develop a plan to develop the mobile applications iteratively
    - Start with the dead bodies identification app; then work on the missing persons app
    - Provide the mock-ups of the apps
    - Develop the UI then start working on the requirements
- Guide A through the mobile application development process
- A asks feedback from his co-workers, which most of them were in the participatory design workshop detailed in Chapter 4
- A changes the mobile application according to the responses at each stage of the development
- Provide a mobile application development methodology for A to follow
- Record A's progress and observe how A responses on the development methodology
- A will have an on-site interview at MIT Computer Science and Artificial Intelligence Laboratory (CSAIL) to share his experience

5.2 Experiment Timeline

Below is the list of our weekly goals -

- Week 1 - Provide mock-up UI and requirements for the two apps
- Week 2 - UI Development for the *Dead Bodies Identification Form*
- Week 3 - Development for the *Dead Bodies Identification Form*
- Week 4 - UI Development for the *Missing Persons Form*
- Week 5 - Development for the *Missing Persons Form*
- Week 6 - Integration with the backend servers.
- Week 7 - A's visit to MIT CSAIL for on-site interview

5.3 Important Scenarios Encountered in the Disaster Relief Efforts

The participatory design workshop and interviews described in Chapter 4 highlight the usefulness of using the Punya framework in a humanitarian context. Below are two apps identified from the workshop and interviews:

- Dead Bodies Identification app (DBI): DBI app present a form for any ICRC worker or other authorized person to record the discovery of human remains. This is done so that the discovery can be documented and later possibly the remains can be cross checked for identification.
- Missing Persons app (MP): MP app presents a form for an ICRC worker or other authorized person to conduct an interview with someone reporting a missing person (friend, relative, etc.), and document the missing person's details.

5.4 Initial Methodology for Guiding Mobile Application Development

I presented several solid design and code principles to A to supplement the development process. The general principle is to plan first before you write the code. The idea is that people should know exactly what they are going to do before they take action during software development.

I took the best practices in software engineering lecture from University of Texas in Austin as a starting point and modified the rules based on the need for mobile application development [14]. The initial methodology below is not comprehensive and it doesn't adequately address all cases. An updated and improved version of methodology will be presented after the experiment. Here were the more detailed rules I shared with A earlier:

1. Model the software visually
    a. Mock-up the user interface
    b. Capture the structure and behavior of components
2. Manage requirements
    a. Reach consensus with colleagues on what the system should and should not do
    b. Understand that user requirements evolve over time and adapt to change in needs
2. Develop iteratively
    a. Set and manage objective milestones
    b. Have initial iteration for early user feedback; continuous testing and integration
3. Verify quality
    a. Develop test suites/strategies to verify functionalities
    b. Diagnose and fix bugs in the app

I didn't force A to strictly follow all the principles, but I constantly reminded him what was stated in the list. I wanted to know if he could understand these principles and whether he could apply them while building the two mobile apps. I observed A while he was building the mobile apps and took notes along the way. I never intervened unless he asked me for help.

5.5 Development Process

This section describes the process of translating and converting the mobile app ideas to mock-ups and then putting the mock-ups together to form the actual user interface. We have the working prototype after this development process.

5.5.1 Mobile App Idea to Mock-ups

We broke the development process down into several stages. First, I asked A to describe and analyze the two apps mentioned earlier. The goal was to have a written form of the two mobile apps. Once he finished this task, I asked him to provide the initial mock-ups for the two apps.

It took him about a week to create the mock-ups, which were done on paper. The mock-ups are shown in Figure 5-1 to Figure 5-5. After the mock-ups were created, A listed all necessary fields needed in the form as shown in Table 5-1 and Table 5.2.

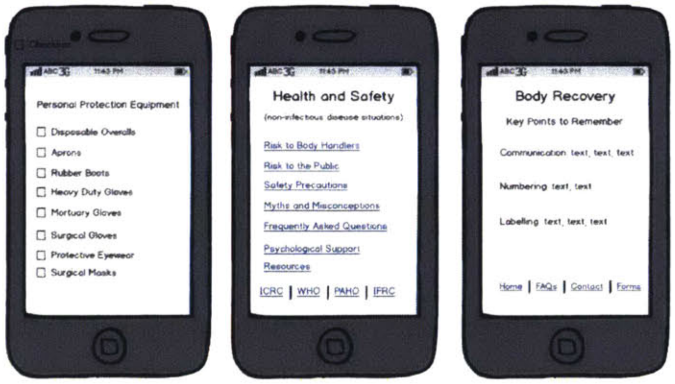


Figure 5-1: Dead Bodies Identification app UI mock-up instruction menu

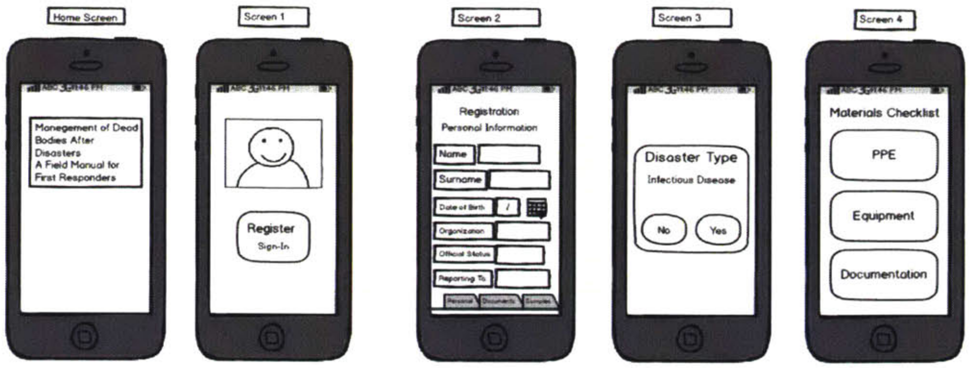


Figure 5-2: Dead Bodies Identification app UI mock-up app user registration pages

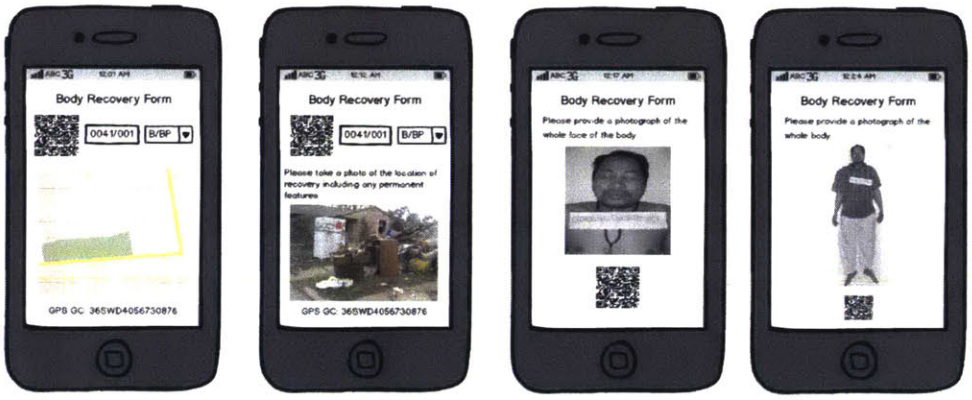
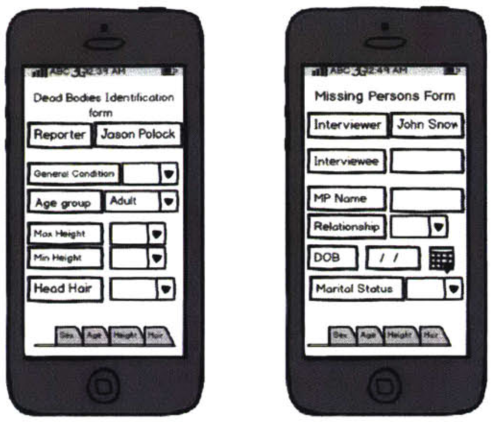

Body Recovery Form
0041/001
B/BP
GPS GC: 36SWD4056730876
Body Recovery Form
0041/001
B/BP
Please take a photo of the location of recovery including any permanent features
GPS GC: 36SWD4056730876
Body Recovery Form
Please provide a photograph of the whole face of the body
Body Recovery Form
Please provide a photograph of the whole body

Figure 5-3: Dead Bodies Identification app UI mock-up reporting pages


Dead Bodies Identification form

Reporter: Jason Polock

General Condition

Age group: Adult

Max Height

Min Height

Head Hair

Missing Persons Form

Interviewer: John Snow

Interviewee

MP Name

Relationship

DOB: / /

Marital Status

Figure 5-4: Personal information collection pages (mock-up)
left: for dead bodies identification form
right: for missing persons form

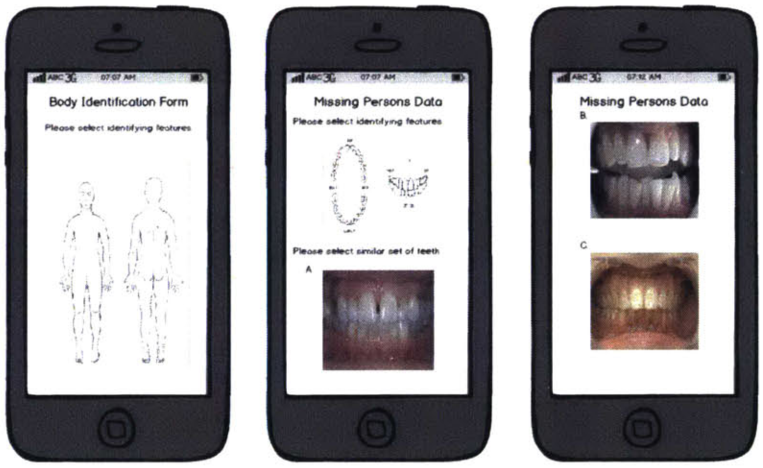

Body Identification Form
Please select identifying features
Missing Persons Data
Please select identifying features
Please select similar set of teeth
A
Missing Persons Data
B
C

Figure 5-5: Body characteristics information collection page (mock-up)
left: human body map for pinpoint distinct features
center and right: take pictures about the distinct features in teeth

Field	Type
Body/ Body part (B/BP) Code	Auto-generated by the app following predefined format

Possible identity of body	Text
Person reporting	Automatically picked up by the app, also contained in the auto-generated code above
Recovery details	Text
Location	Auto-generated by the app
Photograph	<option to take photo or upload one from the system>
General condition	Dropdown {1,2,3,4,5,6}
Apparent sex	Dropdown {m,f}
Apparent sex (evidence)	Text
Age group	Dropdown {0-18, 18-35, 35-55, 55+}
Age group (details)	Text
Height	Number field or dropdown
Weight	Number field or dropdown
Head hair	Dropdown with option "other"
Facial hair	Dropdown
If "Beard" Color	Text or dropdown {black., gray, brown, blond}
If "Beard" Length	Text or dropdown {short, medium, long}
Body hair	Text
Distinguishing features	Text, with option to take photo
Clothing	Text, with option to take photo
Footwear	Text, with option to take photo
Eyewear	Text, with option to take photo
Personal items	Text, with option to take photo
Fingerprints	(Optional)
Identity documents	Text, with option to take photo
Hypothesis of identity	Text
Stored	Text (or dropdown + "other"?)
Responsibility	Text (or a dropdown if list of names of responsible people is available? Needs to be explained.)
Released	<<Functionality to be understood and discussed>>

Table 5-1: Form fields need in the Dead Body Identification app

Field	Type
Missing person number / code	Auto-generated by the app
Interviewer name and contact details (unique info/code of the interviewer)	Automatically picked up by the app, also contained in the auto-generated code above
Place and date (no date needed)	Automatically generated
Interviewee name	Text
Relationship	Dropdown / Text
Address	Text
Telephone	Telephone
Email	Email
Contact name (if different from interviewee)	Text
Relationship (as above)	Text
Address (as above)	Text
Telephone (as above)	Telephone
Email (as above)	Email
Missing person's name	Text
Address	Text
Marital status	Dropdown
Sex	Dropdown
(If female) Unmarried name	Text
(If female) Pregnant?	Y/N
(If female) Children?	Y/N
(If Yes Children) How many?	Dropdown
Date of birth	DOB
Place of birth	Text
Nationality	Text or Dropdown
Principal language	Text or Dropdown
Identity document	Text with option to take photo
Occupation	Text
Religion	Text or Dropdown
Circumstances of disappearance	Text
Has disappearance been reported elsewhere?	Y/N
If yes, explain	Text
Other related missing family members, if any	Text
Height	Number field or dropdown
Weight	Number field or dropdown

Ethnic group	Text or Dropdown
Skin color	Text or dropdown
Eye color	Text or dropdown
Head hair	Dropdown with option "other"
Facial hair	Dropdown
If "Beard" Color	Text or dropdown
If "Bear" Length	Text or dropdown
Body hair	Text
Distinguishing features	Text with option to take photo
Dental condition	Text, with potentially a graphical tool to mark missing teeth
Clothing	Text, with option to take photo
Footwear	Text, with option to take photo
Eyewear	Text, with option to take photo
Personal items	Text, with option to take photo
Identity documents	Text, with option to take photo
Habits	Text
Doctors, medical records, X-rays	Text, with option to take photo
Photographs	Take photo or upload from system
Interviewee signature	Signing on the screen, or maybe an audio clip where they verbally allow the use of the data.

Table 5-2: Form fields need in the Missing Persons app

We finalized the mobile app development plan once we had the final mock-ups and all the fields needed in the forms. A in conjunction with his co-workers (they attended the participatory design workshop previously), came up with all the mobile app requirements.

We agreed to develop the mobile apps iteratively in these three stages: (1) minimum viable product (MVP), (2) MVP and advanced features, and (3) working product. At the minimum viable product stage, we wanted to have a form which captured all the fields. We gradually added the advanced features, such as a login page, submission pages, and the ability to work both online and offline once we had the MVP. Lastly, we wanted to add the features to block mobile app users from uploading data including photos on social media or storing data on a personal device. We also wanted to extend the mobile application to support internationalization. Here is the breakdown of the time spent on each stage:

1. Minimum viable product (MVP)
    a. dead bodies identification app
        i. 3 hrs
    b. missing persons app
        i. 1.8 hrs
2. MVP and advanced features
    a. dead bodies identification app
        i. 5.2 hrs
    b. missing Persons app
        i. 2.8 hrs
3. Working product
    a. dead bodies identification app
        i. 7.5 hrs
    b. missing persons app
        i. 6.2 hrs

Each stage consisted of a teacher-student dialogue. A came to me and asked for help, but a direct solution was never given. Instead I provided hints to help narrow down the bug to a particular section of the code for example when we were in the debugging session. I never help him fix the bug directly.

5.5.2 Mock-ups to Actual User Interface

A can model the features and user interface of the mobile apps visually on a paper. A translated the mobile app ideas to mock-ups. From the mock-ups, he turned them into mobile application user interfaces using the app-building framework. The detailed finished UI are shown in Figure 5-6 and 5-7. Here is the list of the components A identified the app must have.

- labels and textboxes for showing and capturing information, such as text and images
- buttons for advancing to the next page

- google map for finding the current location
- a canvas to collect signatures for release consent form; the canvas component enables people to sign on the mobile phone screen
- a notifier for showing message to users

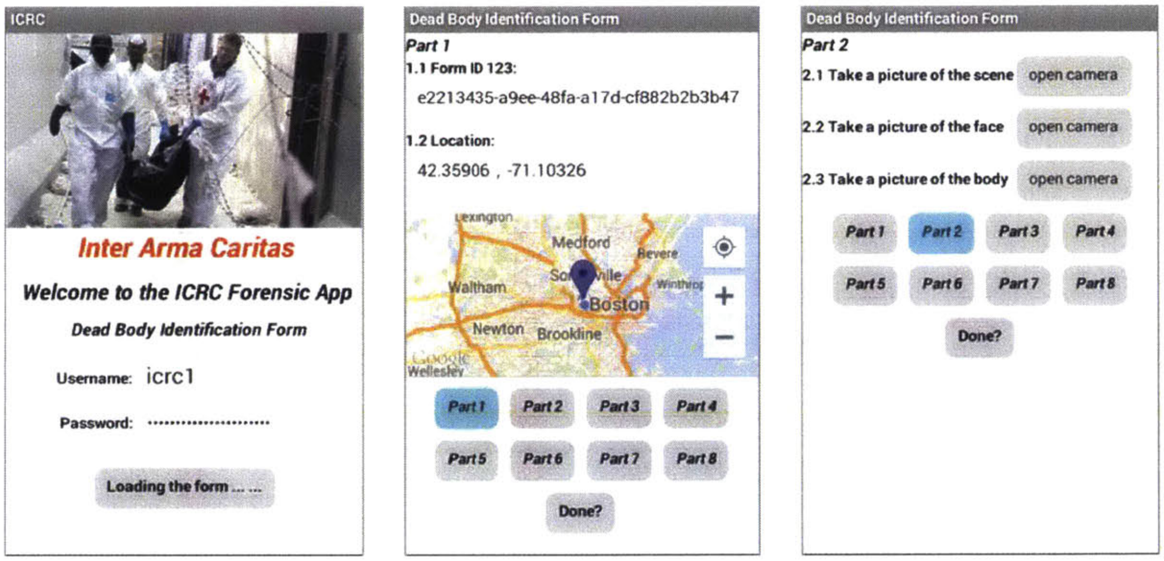
A took the list of the components and then defined the structure and behaviors of each component. A was able to code this up using the Punya framework. Below is the list of the behaviors of the two apps -

- people log in into the app in order to access the form.
- the app automatically assigns the form a unique uuid.
- people can pinpoint their location on the map, or the google map can find user's current location.
- people click a textbox to enter information, or click a button to take pictures.
- people need to submit images first before submitting the form data to the server.
- people submit the form data to the server after images are delivered.
- a notifier will pop up and inform users of app to delete form data if it got delivered to the server successfully as shown in the right image of Figure 5-7.
- users need to delete the form in order to exit the app; the data will be deleted regardless.

A was able to accomplish all the tasks and create the Dead Body Identification app and the Missing Person app. There were many requirement changes, which resulted in numerous modifications to the user interface and form fields. The final working products met all of our requirements. As shown in Figure 5-6 and Figure 5-7, the user interface is differ from the original mock-ups created for the Dead Body Identification app. This is the case for the Missing Person app in Figure 5-9 and Figure 5-10.

The first screenshot in Figure 5-6 is the login screen. The login credential is provided beforehand from the manager of the app. The second screen records the form information, such as form id and report location. The form id is automatically generated and it is unique. For the report location, people need to either pinpoint the location on the Google Map or the ask the GPS component in the phone to find their current location. The third screen in the Figure 5-6 asks people to take pictures of the scene, death body, etc. The first screenshot in the Figure 5-7 asks for more detailed information. The second screen instructs people to submit the form to the server. The third screen shows a successful message which indicates the data is submitted to the server. Users of the app need to delete the form data locally.

The Missing Person app was developed in a way that supports internationalization and we were able to localize the Missing Person app to China. I provided the Chinese translations for the app. The screenshots of the Chinese app are shown in Figure 5-8.


ICRC

Inter Arma Caritas

Welcome to the ICRC Forensic App

Dead Body Identification Form

Username: icrc1

Password: ······················

Dead Body Identification Form

Part 1

1.1 Form ID 123:

e2213435-a9ee-48fa-a17d-cf882b2b3b47

1.2 Location:

42.35906 , -71.10326

Dead Body Identification Form

Part 2

2.1 Take a picture of the scene

2.2 Take a picture of the face

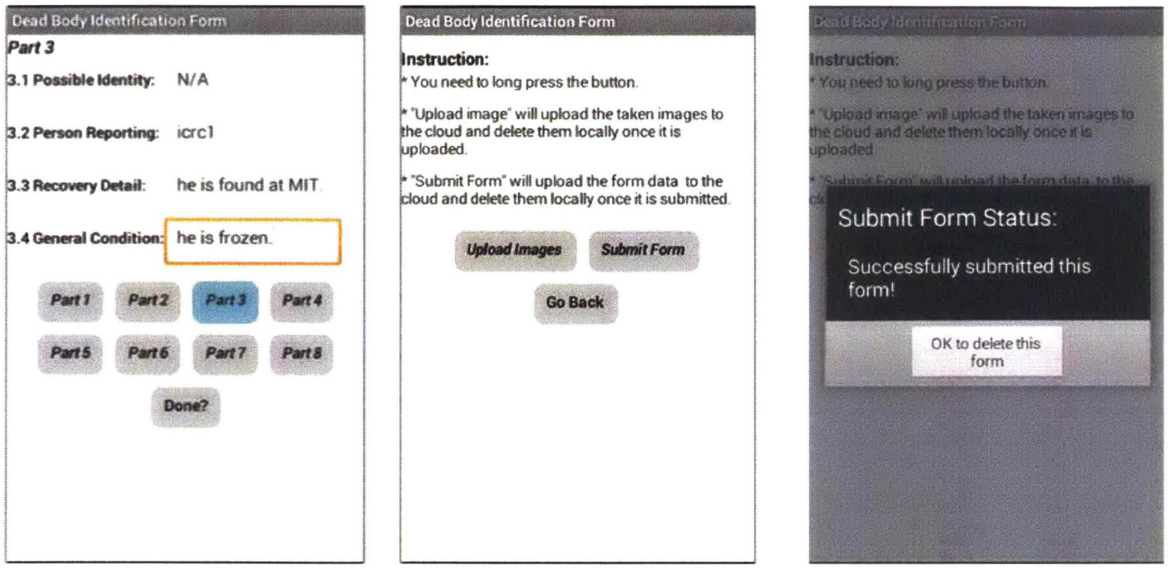
2.3 Take a picture of the body

Figure 5-6: Dead Body Identification app UI

left: login page with predefined credentials

center: auto form id generation and pinpoint geo-location

right: page for taking pictures, such as scene, face, and body



Dead Body Identification Form

Part 3

3.1 Possible Identity: N/A

3.2 Person Reporting: icrc1

3.3 Recovery Detail: he is found at MIT.

3.4 General Condition: he is frozen.

Dead Body Identification Form

Instruction:

* You need to long press the button.

* "Upload image" will upload the taken images to the cloud and delete them locally once it is uploaded.

* "Submit Form" will upload the form data to the cloud and delete them locally once it is submitted.

Submit Form Status:

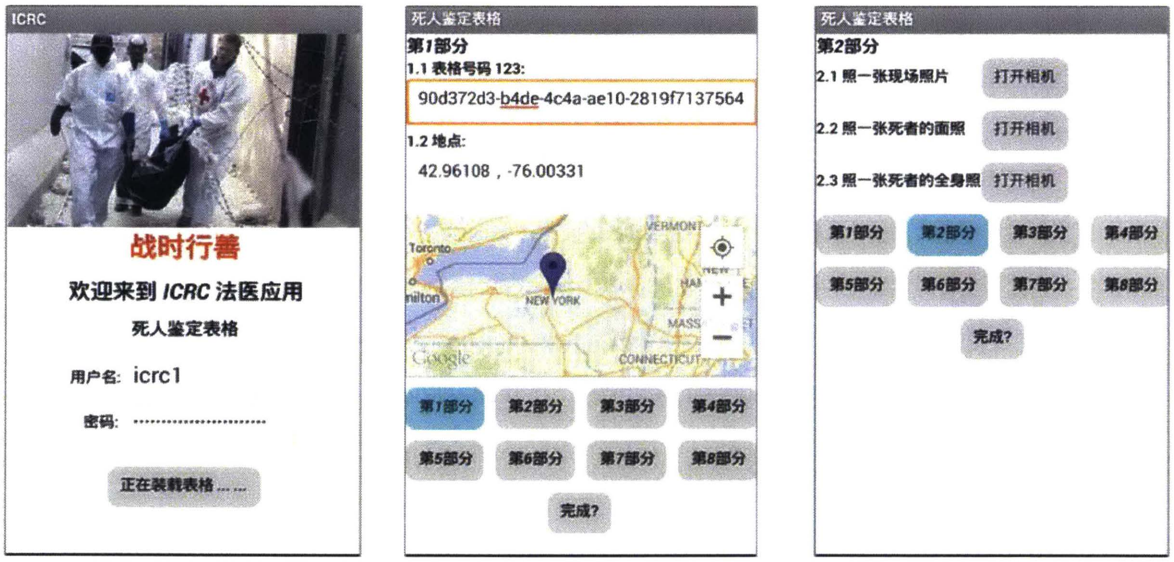
Successfully submitted this form!

Figure 5-7: Dead Body Identification app UI

left: page for collecting personal information

center: submission page with instruction

right: message box for successfully submission



ICRC

战时行善

欢迎来到 *ICRC* 法医应用

死人鉴定表格

用户名: icrc1

密码: ························

死人鉴定表格

第*1*部分

1.1 表格号码 123:

90d372d3-b4de-4c4a-ae10-2819f7137564

1.2 地点:

42.96108，-76.00331

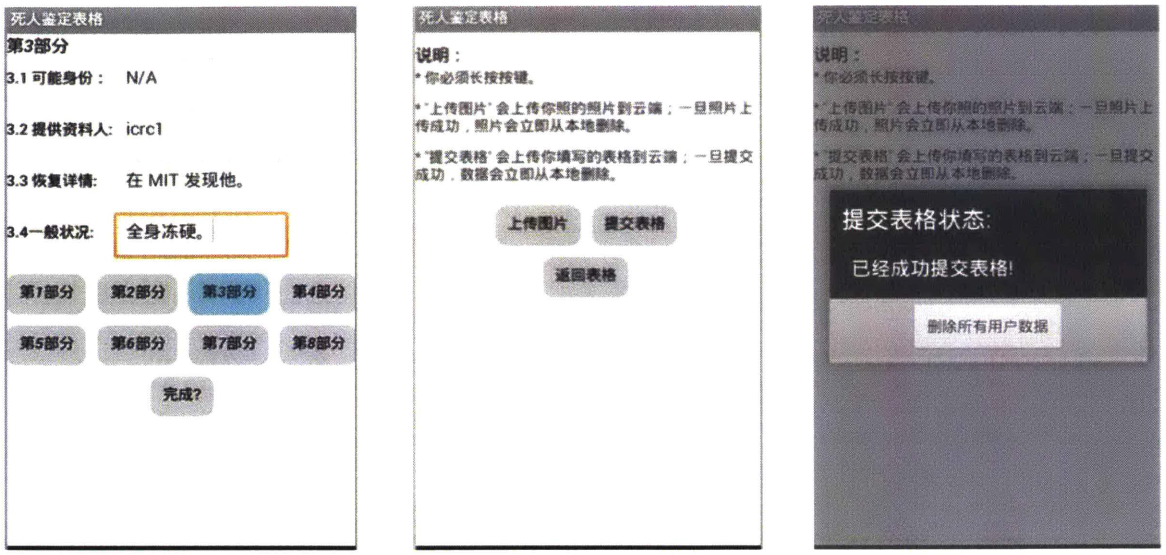
死人鉴定表格

第2部分

2.1 照一张现场照片

2.2 照一张死者的面照

2.3 照一张死者的全身照



第3部分

3.1 可能身份： N/A

3.2 提供资料人: icrc1

3.3 恢复详情: 在 MIT 发现他。

3.4一般状况: 全身冻硬。

说明：

* 你必须长按按键。

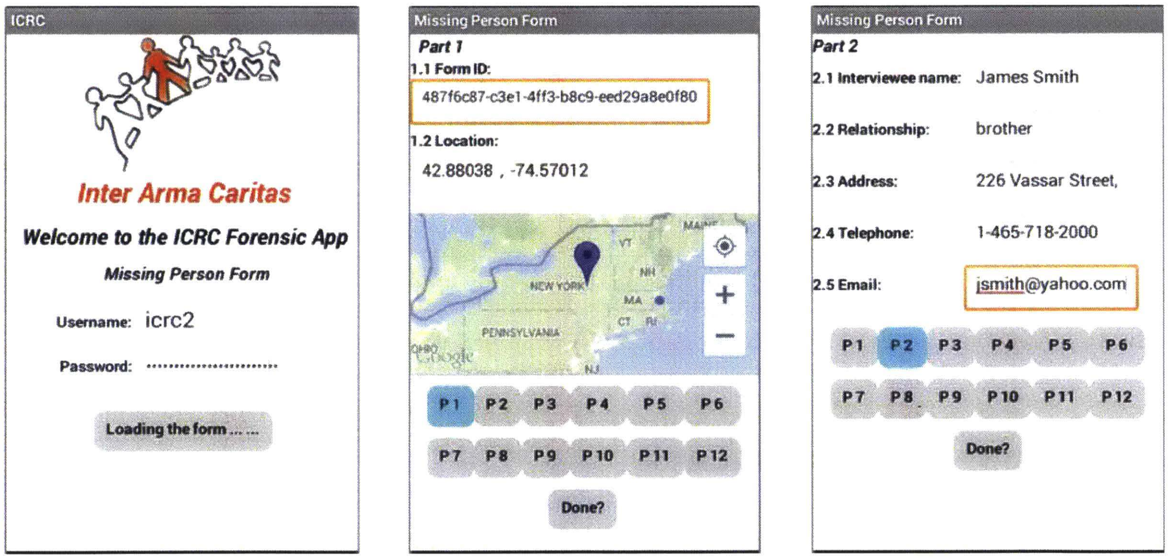
* "上传图片" 会上传你照的照片到云端；一旦照片上传成功，照片会立即从本地删除。

* "提交表格" 会上传你填写的表格到云端；一旦提交成功，数据会立即从本地删除。

提交表格状态:

已经成功提交表格!

Figure 5-8: Dead Body Identification app in Chinese



Welcome to the ICRC Forensic App

Missing Person Form

Username: icrc2

Password: ························

Part 1

1.1 Form ID: 487f6c87-c3e1-4ff3-b8c9-eed29a8e0f80

1.2 Location: 42.88038 , -74.57012

Part 2

2.1 Interviewee name: James Smith

2.2 Relationship: brother

2.3 Address: 226 Vassar Street,

2.4 Telephone: 1-465-718-2000

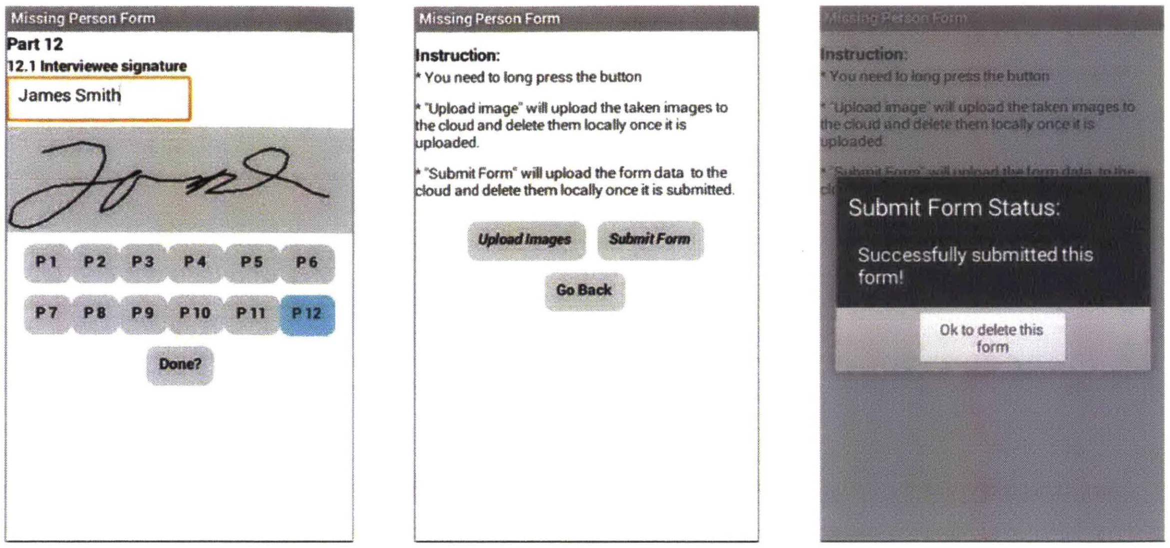
2.5 Email: jsmith@yahoo.com

Figure 5-9: Missing Person app UI

left: login page with predefined credentials

center: auto form id generation and pinpoint geo-location

right: page for collecting personal information


Missing Person Form

Part 12

12.1 Interviewee signature

James Smith

Missing Person Form

Instruction:

* You need to long press the button

* "Upload image" will upload the taken images to the cloud and delete them locally once it is uploaded.

* "Submit Form" will upload the form data to the cloud and delete them locally once it is submitted.

Submit Form Status:

Successfully submitted this form!

Figure 5-10: Missing Person app UI

left: Interviewee signature collection page for authorization

center: submission page with instruction

right: message box for successfully submission

The screenshots of the missing person app UI are shown in Figure 5-9 and Figure 5-10. The UI is very similar to the dead body identification one. The missing person app has an additional feature for digital signature collection as shown in Figure 5-10.

5.6 Manage Requirements

A realized the need for some changes to the app. I gave out advice on the UI improvements occasionally and A kept his original idea most of the time. Here is the an example where A came up with UI draft and then it was modified as a result of my interactions in the UI. He wanted the interviewee to sign the release form on paper and then take a picture for digital record. I recommended that he follow the digital doodle tutorial which lets users draw freely on the mobile screen. He reported this idea to ICRC forensic team and had a discussion with them whether they should solve this issue as I had suggested. A was able to reach a consensus with the ICRC forensic team and used this solution. A was able to handle the requirement changes when they arose during the development process. Figure 5-11 shows the final working product screenshot.

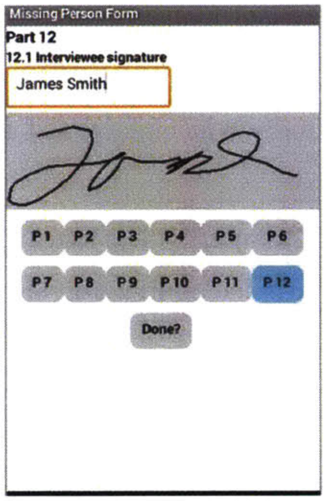


Figure 5-11: Digital signature feature in the missing person form app

5.7 Develop Iteratively

Developing the mobile application iteratively helped A a lot during the mobile app development, especially during setting, managing, and meeting milestones. A felt overwhelmed at the beginning because he didn't have the necessary experience. I explained to A that he could start from the most critical part first as the minimum viable product, and then expand features from the minimum viable product. A was able to break down the mobile application into three stages and each stage was built on top of the previous one. Here is the list of features in each stage -

- Minimum Viable Product

    - build the form to capture the text information
    - have the ability to take pictures using the camera and store them afterward
- MVP and advanced features
    - build the login screen and authenticate users before they use the form
    - be able to upload the images and the data in the form to the server
- Working product
    - use the canvas to capture the digital signature
    - delete the image and text data immediately after the form has submitted to server
    - arrange the mobile screens in an intuitive and user friendly way

At each development iteration, he invited users to test that stage and asked for early user feedback. This process helped him focus and prioritize during the task accordingly.

5.8 Verify Quality

It was difficult for A to verify quality of the mobile applications. In particular, quality verification means to develop test cases used to validate the functionalities. A was not able to diagnose and fix bugs in the mobile applications.

When putting together the test suites strategies, professional developers utilize everyday use cases and rare use cases. A could come up with the everyday use cases, but failed to think of the outliers (rare) uses. For example, A was able to follow the instruction to submit the form successfully to the server; but he could come up with a test case such as the phone stop when the user tried to enter the data into the form and the intermediate data was lost as a result of that. A wasn't satisfied with this behavior and he redesigned the internal logic to capture all the intermediate data afterward. The reason of causing this is the lack of the professional training. A didn't go through the training of coming up corner and rare use cases to test the software. This kind of training is common in a lot of the software engineering programs.

Debugging was a challenge for A. Sometimes, he could not locate, or identify the problems or bugs in the code. A told me it was very difficult to locate specific blocks because of the number of blocks present in the workspace. The limited amount of workspace made it even harder to find blocks which presented bugs. A also outlined that the lack of a search function in the block workspace hindered his productivity. He was familiar with the search function normally used within other frameworks and he was expecting the same feature to be available in the app-building framework.

5.9 Applying Linked Data

Disaster management activities generate data. In particular, coordinating relief operations, reporting structural damages, harms, death, and overseeing volunteers produces a lot of data. This data needs to be integrated and be interoperable with various organizations in order to be useful.

A was taught the Linked Data can help with data integration and data interoperability, and he used Linked Data in these two applications. A was able to understand the concepts behind the Linked Data form, Linked Data ontology, and Linked Data technology after he was given several detailed Linked Data instructions. The Linked Data form is a layout component that can be used by app developers to identify a collection of fields. These fields are applied to a particular ontological concept. Developer can add fields to the Linked Data form anytime. Linked Data forms can also assign an auto-generated, common subject URI to the field.

One challenge of using Linked Data is finding details of an ontology that describes a dataset. Because the lack of necessary experience, A couldn't supply any ontology to the form.

http://xmlns.com/foaf/0.1/familyName

A was supplied with the property uri. He then filled in each field in the dead body identification form and missing person form. For example, the property uri for the family name is the following:

5.10 Key Findings

Overall, I found that even though A was non-technical, A was able to build and deploy the Dead Body Identification app and the Missing Person app after I had provided the proper guidance. I got positive feedback about the experiment from him. The main findings of the experiment are:

1. A can model mobile apps visually. A translated mobile app ideas to mock-ups and then turned them into user interfaces. A captured the structure and behaviors of each component and was able to code it using the Punya framework.
2. A and his co-workers came up with the mobile app requirements and A managed requirement changes well during the development process. A was able to reach a consensus with other ICRC users (forensic team) on what the mobile applications should and should not do. A adapted to the requirement changes alongside the team.
3. Developing the mobile application iteratively helped him set, manage, and accomplish objective milestones. A was able to break down the mobile application into three stages and each stage was built on top of the previous one. At each development iteration, he invited users to test the app and asked for early user feedback. This process helped him focus and prioritize during each the task.
4. A was able to create the Missing Person app by modifying the Dead Body Identification app. Because these two apps share many similarities, in order to create the Missing Person app, he reused the code created for the Dead Body Identification app and edited the contents (mainly adding more fields and changing the field labels).
5. A was able to understand the concepts of a Linked Data form, Linked Data ontology, and Linked Data technology after given five detailed Linked Data instructions. He wasn't

able to supply the property uri to each field in the dead body identification form and missing person form due to the lack of experience.

6. A could not created good test cases to verify quality of the mobile applications since he wasn't trained to do so. A was unable to diagnose and fix bugs mainly due to the lack of block workspace and search function in the Punya framework.
7. A found difficulty in creating app in the Punya framework. There is no copy and paste feature in the framework. When a form with many fields is created , users of the Punya framework need to gather and organize each element manually. Specifically, many of the elements are the same in the form, such as textbook, label, etc. There were many repetitive motions involved in putting together the two forms. A was hoping to use a mechanism in the Punya framework to generate forms automatically. A also wanted to be able to reuse existing forms from the personal form collection.

5.11 Updated Methodology for Guiding Mobile Application Development

Here is the updated version of the methodology that includes the updates from the experiment.

1. Model the software visually
    a. Mock-up the user interface on paper first
    b. *Break the user interface into components*
    c. Capture the structure and behavior of each component and write them down
2. Manage requirements
    a. *Present the requirements to the colleagues in the team early*
    b. Reach consensus with colleagues on what the system should and should not do
    c. Understand that user requirements evolve over time and adapt to change in needs
3. Develop iteratively
    a. *Back up the project on a regular basis*
    b. Set and manage objective milestones

    c. *Identify the minimum viable products and each additional features*
    d. Have initial iteration for early feedback and do testing and integration as well
4. Verify quality
    ~~a. Develop test suites/strategies to verify functionalities~~
    b. *Write test plan on each component to verify functionalities*
    c. *Invite people to test the app early based on the test plan*
    d. Diagnose and fix bugs as soon as you discover it in the app

Point 1.b, 2.a, 3.a, 3.c, 4.b, and 4.c are the new principles and point 4.a is removed. This is not the final version of the methodology. The final one takes the new toolkit that introduces in Chapter 6 into consideration and is more comprehensive than this one. Chapter 7 presents the final methodology for relief workers to follow while creating apps using the new toolkit.

Chapter 6

Architecture and Implementation of the Mobile Linked Data App Kit

This chapter describes the design and implementation of the Mobile Linked Data App Kit. It has three parts, (1) the Punya framework as explained in Chapter 2 of this thesis, (2) the Linked Data Form Extension, a new feature in the Punya framework, and (3) the Virtuoso Docker container for easy Linked Data store deployment. The Linked Data Form Extension includes a Linked Data Form Generator that auto-generates a Linked Data form when an ontology is provided. The form generator also generates the appropriate SPARQL query enabling the developer to take advantage of structured data without understand Linked Data concepts. It is both for data collection and data querying. The generator also includes a Duplicate Screen Feature created to copy and paste existing screens and a Share Screen Feature that allows importing and exporting screens within a project. The Virtuoso Docker container brings up an instance of the Virtuoso Linked Data store within minutes using only a few simple commands.

Screen in the framework means the whole view or the single user interface in the mobile application. A form is a subset of the screen which specialize in collecting input data. A screen can contain multiple forms. In general, a form contains a set of of text labels and textboxes. As shown in Figure 6-1, the screen is the whole view and the Linked Data form is a subset of the screen. Linked Data form is a data type and it contains many elements, such as labels, textboxes, etc in it.

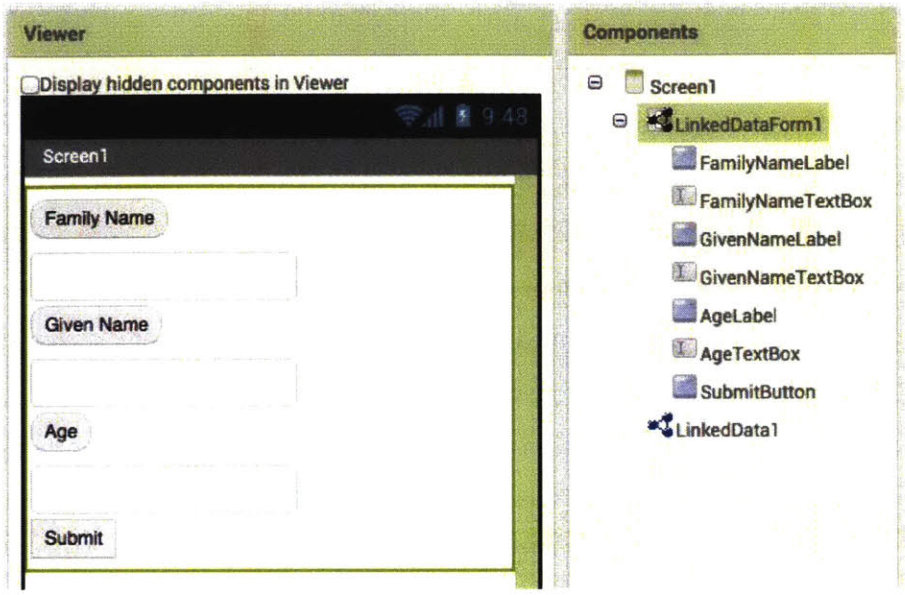


Figure 6-1: A Linked Data Form in Screen 1

The Linked Data Form Extension can be used by people to easily build apps with a Linked Data form. Specifically, the new feature allows people to build Linked Data forms automatically when given an ontology in the screen. The new feature provides the ability to copy existing screen and to share an existing screen, which include the Linked Data form.

The chapter begins with the motivation behind having the Mobile Linked Data App Kit. Next, the chapter presents an overview of the generator to highlight the roles and functions of the different major parts. The chapter then introduces implementation details of the sub-modules within each part. Lastly, the chapter will introduce an easy Linked Data store deployment plan.

6.1 Motivation

Using Linked Data in mobile applications has been shown to be useful in the humanitarian field in the participatory design workshop and interviews in Chapter 4. The Punya framework provides people with an easy way to assemble a mobile Linked Data app. From my own experience from many Punya workshops, early adopters of the Punya framework reported creating a Linked Data form in the Punya framework was not intuitive and somewhat difficult. As stated in section 4.5 of Chapter 4, the ICRC workshop participants explained that the lack of the copy and paste function made it harder to create Linked Data forms as they needed to repeatedly drag and drop the same form elements even when they were essentially the same between the two forms. Furthermore, they complained that they couldn't reuse the screens that they had built earlier, or they had obtained from someone else, since sharing screens, either within the same accounts, or from different accounts was not possible. Lastly, I conducted a small workshop during the 12th European Semantic Web Conference and the early adopters demand an easier deployment plan for the Linked Data store. Current Linked Data Stores, such as Virtuoso, Mulgara, Sesame Native, Jena SDB, and many other triplestores, require advanced IT skills to deploy and use.

The above mentioned issues motivated the creation of the Mobile Linked Data App Kit. We want to help people to create and deploy humanitarian relief mobile apps using the kit. Eventually, they can utilize these apps to assist their humanitarian work

6.2 Ideal Scenario Using the Mobile Linked Data App Kit

In the aftermath of a hurricane, B who is the project manager from Organization C want to build a mobile application targeted at humanitarian workers on the ground. The app needs to collect details of missing persons. B needs to have the app collect many biometric information, similar to the ICRC *Missing Person App* outlined above in Chapter 4. B also wants to add the capability to subscribe to missing person reports. This capability is similar to the subscription feature in the *WeReport* app which I outlined in Chapter 3.

B needs to deploy a triplestore for storing Linked Data. B has only very limited time given the disaster situation. B can't build the app from scratch. Ideally, B would like to use the form in the *Missing Person App* as a template and modify it with fields needed in the form. B also needs to add the additional feature from the *WeReport* app. He needs to create a new mobile app on the fly. B also needs to create a triplestore which should be compatible with the new mobile app.

The above scenario is the kind of real life situation that could benefit from using the Mobile Linked Data App Kit. B can load the existing *Missing Person App* using the Punya framework, add more fields using the Linked Data Form Extension. B can import the subscription feature from the *WeReport* app using the Screen Import/Export feature, and deploy a triplestore instance immediately using the Virtuoso Docker container.

6.3 Overview

The Linked Data Form Extension is built on top of the Punya framework. As explained in Chapter 2, The Punya framework is an app building platform containing components which can be used to build mobile Linked Data apps. The Linked Data Form Extension consists of three parts: (1) a Linked Data Form Generator that auto-generates a Linked Data form when an ontology is provided, (2) a Duplicate Screen Feature created to copy and paste existing screens and (3) a Share Screen Feature that allows importing and exporting screens within a project.

6.3.1 Workflow of Linked Data Form Generator

This section describes the main workflow of the Linked Data Form Generator. People need to log into the app building platform. The generator is a new extension to the Punya framework. Once users are logged in the app building platform, the workflow happens in the following steps:

1. People create a new project or select an existing project
2. The Linked Data Form Generator is shown as a button named *Generate LD Form* on the upper panel in the designer view.
    a. People can click on this button to select such an option.
    b. A dialog box pops up and asks people to supply an ontology

    c. Once people supply an ontology, the generator will populate all the fields in the ontology.
3. A form will be generated with the selected field in the designer viewer.
    a. The corresponding settings, such as field property url and blocks will be generated in the background as well.
4. A collection of blocks for submitting the form to a triplestore will be generated in the block editor.
5. A collection of blocks for SPARQL query will be generated in the block editor as well.
    a. This SPARQL query is matched with the generated form and query all the data came from this form.
    b. The query is built with the fields selected from the provided ontology.
    c. The query limits the result to only the first 100 entries.
6. People can generate multiple forms in the same screen or view in the designer

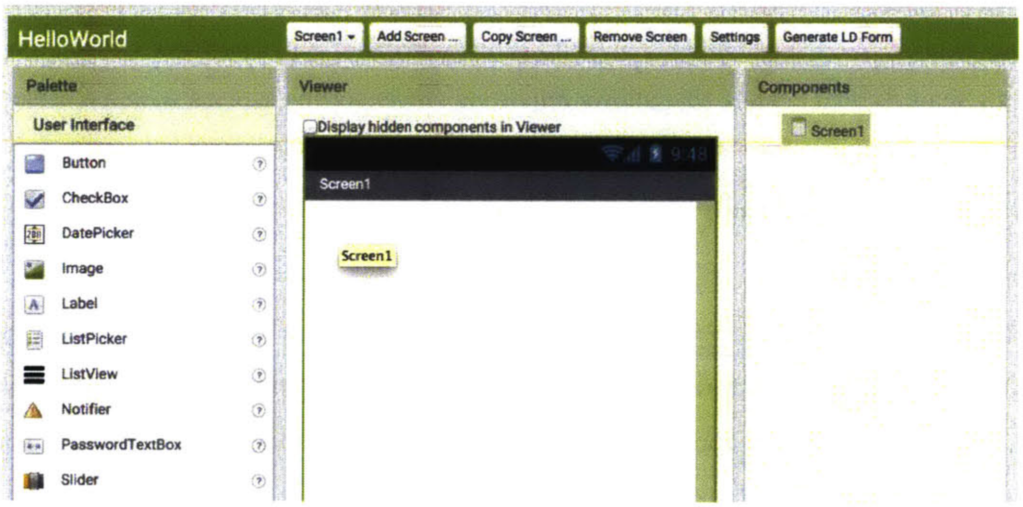
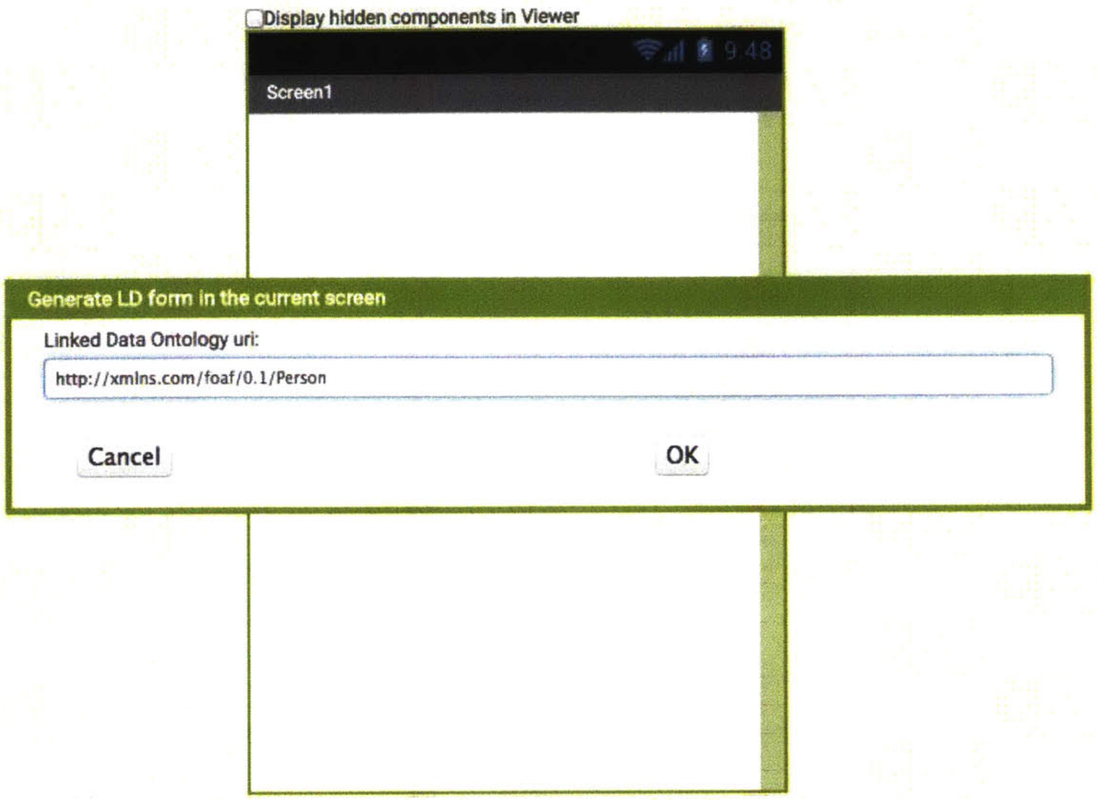
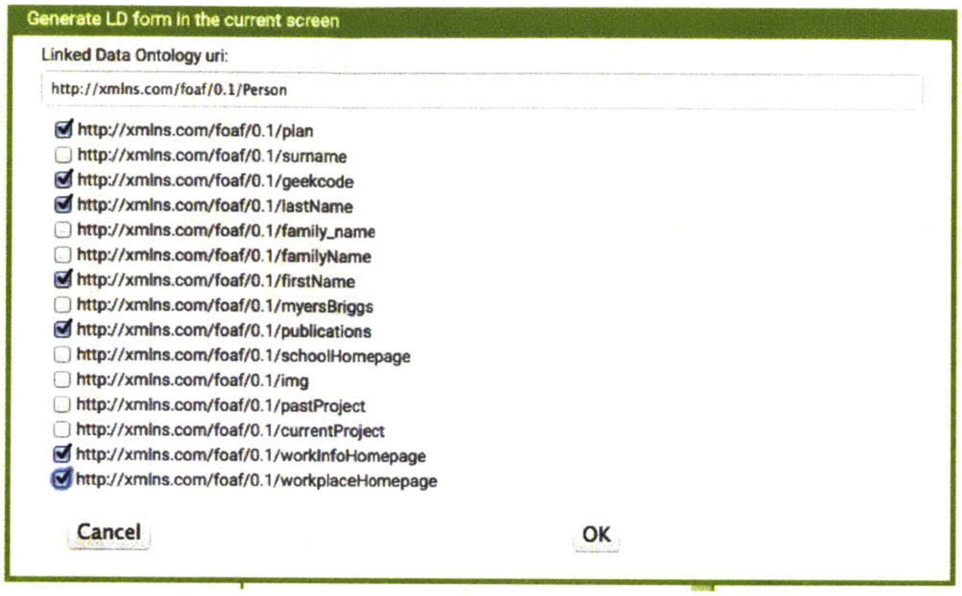


Figure 6-2: Designer layout with the "Generate LD Form" option new feature



Display hidden components in Viewer
9:48
Screen1
Generate LD form in the current screen
Linked Data Ontology uri:
http://xmlns.com/foaf/0.1/Person
Cancel
OK

Figure 6-3: Linked Data Form Generator UI asking for an ontology;
a FOAF/Person ontology is supplied as shown

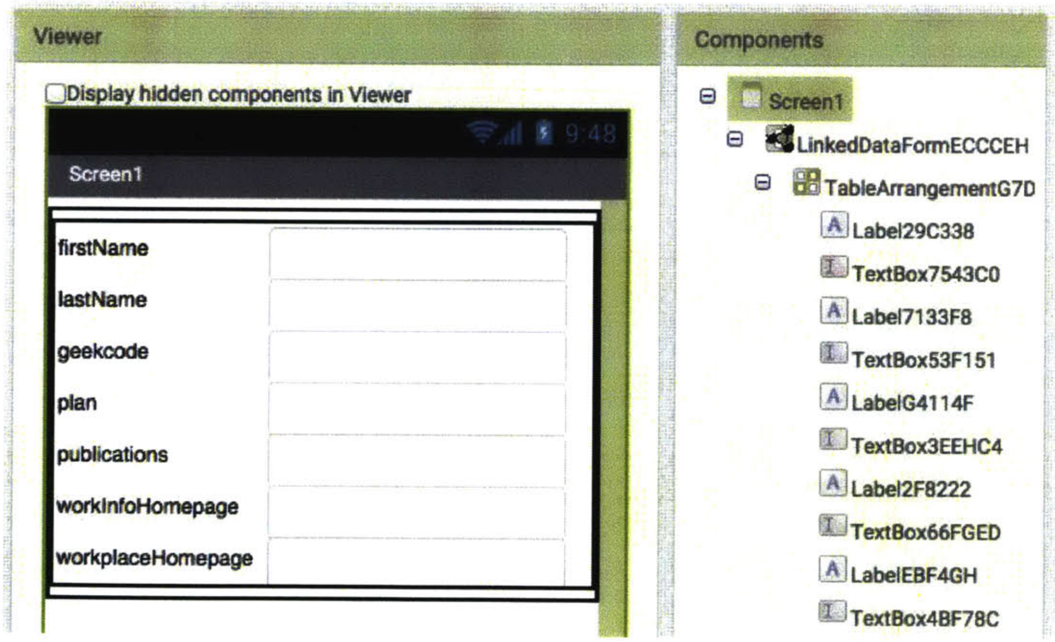

Generate LD form in the current screen

Linked Data Ontology uri:

http://xmlns.com/foaf/0.1/Person

- [x] http://xmlns.com/foaf/0.1/plan
- [ ] http://xmlns.com/foaf/0.1/surname
- [x] http://xmlns.com/foaf/0.1/geekcode
- [x] http://xmlns.com/foaf/0.1/lastName
- [ ] http://xmlns.com/foaf/0.1/family_name
- [ ] http://xmlns.com/foaf/0.1/familyName
- [x] http://xmlns.com/foaf/0.1/firstName
- [ ] http://xmlns.com/foaf/0.1/myersBriggs
- [x] http://xmlns.com/foaf/0.1/publications
- [ ] http://xmlns.com/foaf/0.1/schoolHomepage
- [ ] http://xmlns.com/foaf/0.1/img
- [ ] http://xmlns.com/foaf/0.1/pastProject
- [ ] http://xmlns.com/foaf/0.1/currentProject
- [x] http://xmlns.com/foaf/0.1/workInfoHomepage
- [x] http://xmlns.com/foaf/0.1/workplaceHomepage

Figure 6-4: Linked Data Form Generator UI asking people to select fields for the form; first name, last name, plan, geek code, work info homepage and workplace homepage are selected.



firstName
lastName
geekcode
plan
publications
workInfoHomepage
workplaceHomepage

Screen1
LinkedDataFormECCCEH
TableArrangementG7D
Label29C338
TextBox7543C0
Label7133F8
TextBox53F151
LabelG4114F
TextBox3EEHC4
Label2F8222
TextBox66FGED
LabelEBF4GH
TextBox4BF78C

Figure 6-5: Linked Data Form Generator UI asking to select fields for the form; first name, last name, plan, geek code, work info homepage and workplace homepage are selected

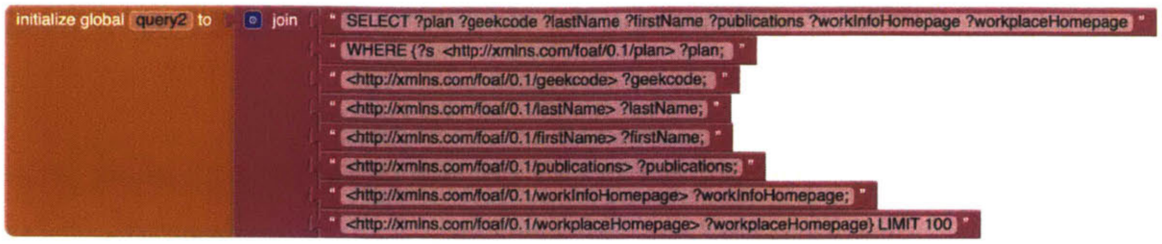
A finished FOAF/Person form is shown in Figure 6-5. People can build the form without the generator, but people would need to manually build the form by pulling the buttons, labels in the form. It is a time consuming and repetitive process to do without the generator. The Linked Data Form Generator automates and simplifies this process.

The generator provides the available ontologies. There are currently three ontologies available for experimental use at this points and more will be added to the collection later. They are FOAF, HXL, and MOAC, which are essential to the humanitarian assistance. Users need to choose the ontology before selecting the fields they need to use. There is no limit on the number of properties or triples that an ontology can have to work with the generator. However, there is limited space in the phone screen and users of the generator must keep that mind to avoid any usability issue.

All the text fields in the auto generated Linked Data form include the selected property url from the ontology. People can view the property url in the property panel on the right side of the designer panel.

There was one major challenges faced when implementing generator. In the current implementation, multiple instances of the same property with different semantics. There wasn't an easy way to prevent that in the runtime, so I implemented some checking mechanisms in the compiler time. When the Linked Data form in the app has multiple instances of the same property with different semantics, an error message will be popped up to inform the users.

This section illustrates form creation and the Linked Data aspect of this happens in the background. The RDF of the Linked Data form will be produced when the form is requested to submit to the datastore. Figure 2-8 shows the necessary blocks needed to submit the RDF to the remote datastore. This remote datastore is provided / hosted by the third party, not by Punya team. As shown in Figure 6-6, a corresponding SPARQL query is being produced by the Linked Data Form Generator. It lists all the entries in the datastore according to the form.



```
initialize global query2 to join
" SELECT ?plan ?geekcode ?lastName ?firstName ?publications ?workInfoHomepage ?workplaceHomepage "
" WHERE {?s <http://xmlns.com/foaf/0.1/plan> ?plan; "
" <http://xmlns.com/foaf/0.1/geekcode> ?geekcode; "
" <http://xmlns.com/foaf/0.1/lastName> ?lastName; "
" <http://xmlns.com/foaf/0.1/firstName> ?firstName; "
" <http://xmlns.com/foaf/0.1/publications> ?publications; "
" <http://xmlns.com/foaf/0.1/workInfoHomepage> ?workInfoHomepage; "
" <http://xmlns.com/foaf/0.1/workplaceHomepage> ?workplaceHomepage} LIMIT 100 "
```

Figure 6-6: A corresponding SPARQL query is being produced by the Linked Data Form Generator

6.3.2 Copy and Paste Screen Feature

The Punya framework is a blocks-based app building platform. The lack of a copy and paste feature in the designer part of the platform can be counterproductive, especially when people are engaged in repetitive tasks. In order to improve this issues, a T copy and paste screen feature was created as part of the Linked Data Form Extension. The current copy and paste feature works on the screen level, as opposed to, for example, the component level. The workflow happens in the following steps:

1. People open the view of the screen which they wish to make a copy.
2. They click the Copy Screen button and a dialog box will open requesting the new form name.
3. A new screen with the entered name will be generated in the screen background and the page will be refreshed.
4. All the components in the Designer view and the elements in the Blocks Editor view will be cloned to the new screen.
5. The corresponding reference to the old screen will be automatically changed to the new screen.

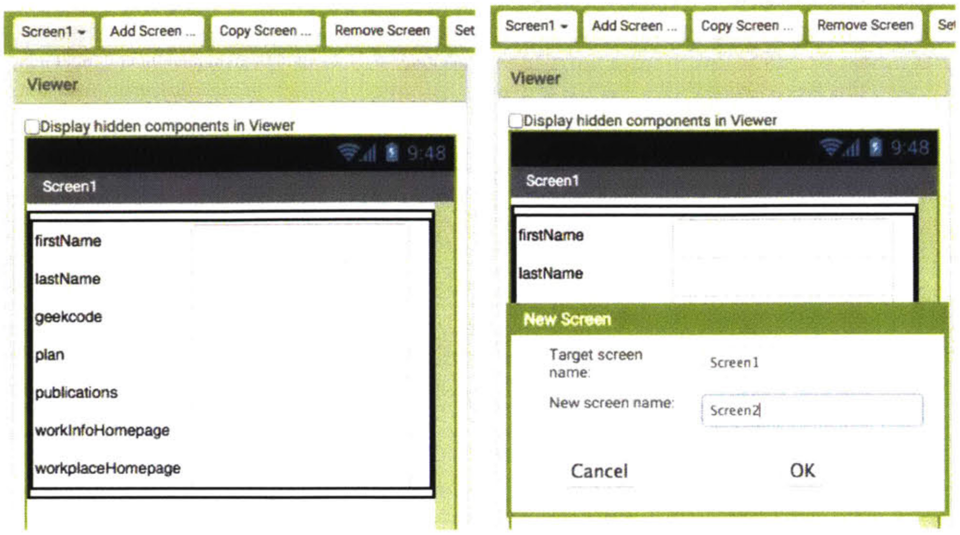


Figure 6-7: (left) The "Copy Screen" option is at the top panel (right) It is a dialog box for naming the new form/screen new feature

6.3.3 Import / Export Screen Feature

People who use the Punya framework often want to reuse what they have built or developed within previous or current project. The Punya framework supports importing an entire project, but not imports of individual screens. For example, Tom is a busy developer and he wants to reuse or transfer the people form he built in project 1 to project 2. Currently, there is no way to do that in the Punya framework without building the people form all over again. However, using the new share Linked Data form feature, Tom can export the whole screen containing the people form in project 1 as a file. Tom can then import this file to project 2. The file will then translate to a new screen which contains a people form. The app development process is more agile now. People are encouraged to share what they have done, to reuse existing resources, to modify the existing resource, and hopefully create their own resources faster.

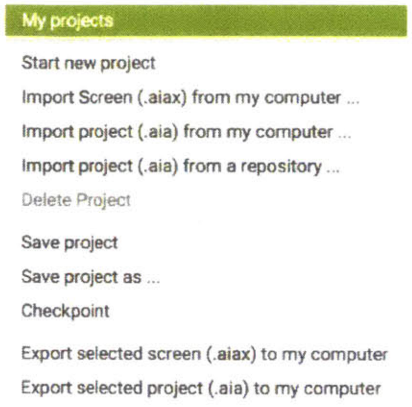

My projects
Start new project
Import Screen (.aiax) from my computer ...
Import project (.aia) from my computer ...
Import project (.aia) from a repository ...
Delete Project
Save project
Save project as ...
Checkpoint
Export selected screen (.aiax) to my computer
Export selected project (.aia) to my computer

Figure 6-8: Option to export the current screen as a file

from the current project

new feature

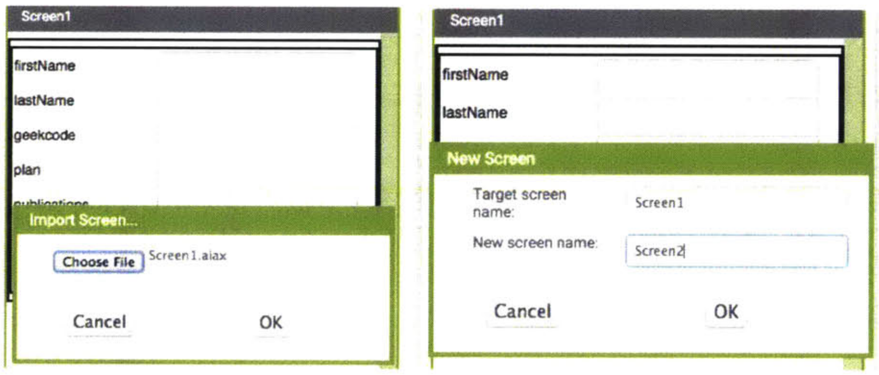


Figure 6-9: Import a screen file to the current project as a new screen

new feature

6.4 Architecture of the Linked Data Form Extension

The Linked Data Form Extension performs two operations. First, the generator needs to produce the screen which contains the elements displayed as part of the user interface in the designer.

Second, it needs to produce the functional blocks which contain the actions in the block editor. This operations happen simultaneously in the background.

In the current implementation of the Punya framework, each element in the screen is modeled as data node. This is also the case for the blocks. Each screen has two lists of data nodes; one list is for the screen elements and the other list is for the blocks. So the projects itself contains all the list of data nodes. If we need to generate the screen, we need to add the collection of data nodes to the project list. There are three cases we need to consider:

(1) Auto generate the Linked Data form given an ontology
(2) Copy and paste existing screens
(3) Exporting and importing screens for reuse

For the first case, the extension adds new data node to the current screen element list and blocks list according to the elements in the ontology. For the second case, we just need to select the particular list of the screen elements and list of the blocks given the selected screen and then make a duplicate list each with a new name. For exporting forms, the extension outputs the screen element list and block list as as text file. Likewise, for importing, the extension takes the text file and translates it into two new lists, one for screen elements, and other one for blocks.

6.5 Easy Linked Data Store Deployment Plan

Linked Data Form Extension simplifies the process of generating and consuming Linked Data, but Linked Data store deployment remains a challenge for many people. The process of deploying a Linked Data store requires sophisticated IT knowledge. Linked Data store compatibility issues often come up when I conducted Linked Data workshop. Having a compatible version of Linked Data store with our Linked Data components is very important for the Mobile Linked Data App Kit. These issues motivated the simplification of this process by utilizing the mature softwares in the market and making necessary customizations.

Virtuoso is the product I use and it provides SQL-Object-Relational Database Management Systems. In particular, I use the RDF data management and the triplestore access via SPARQL API. Virtuoso is the leading provider for Linked Data triplestore and I am currently using the open source free version - *OpenLink Virtuoso.* Docker containers wrap up a piece of software in a complete filesystem that contains everything it needs to run: code, runtime, system tools, system libraries – anything that can be installed on a server. This guarantees that it will always run in the same way, regardless of the environment it is running in.

I have created a Docker container for the Virtuoso and it is part of the Mobile Linked Data App Kit. This Virtuoso Docker container is compatible with our Linked Data components. I have hosted the Virtuoso Docker container online and it is publicly available to download and install. Also, I placed an image of the Linux OS with the Virtuoso Docker container on Internet. People can deploy an instance of the Virtuoso within two clicks once they have the server to host the Linked Data. This reduces a significant amount of barriers to entry and eases the burden of Linked Data store deployment.

6.6 Summary

The Linked Data Form Extension lowers the technical barriers for people to build mobile applications with the Linked Data form in the Punya framework. It makes easier for people to produce and consume Linked Data. Having an easy Linked Data store deployment through the Virtuoso Docker container mitigates people's burden of setting up the infrastructure of a Linked Data store. I will discuss the possible future directions in Chapter 7.

Chapter 7

Evaluation

There are several components to this Do-It-Yourself (DIY) toolkit: (1) Mobile Linked Data App Kit: a streamlined process to create mobile apps and (2) the app-building methodology: a methodology for relief workers to follow while creating apps. For the evaluation part, I am going to focus on the Mobile Linked Data App Kit, which includes the Punya framework, the Linked Data Form Extension, and the Virtuoso Docker container. In particular, I am going to walk through a mobile application development case using a collection of templates.

7.1 Collection of Templates

Over the course of this thesis and many workshops and onsite visits, I have found that many humanitarian organizations have the need to build mobile applications. These mobile applications can potentially help them in the field and assist them in their aid work during a disaster or emergency. When I looked closer into these needs, or the requirements for the mobile applications, I found many of them shared a lot of similarities in term of functionalities. Here is the list of the functionalities found in these mobile applications:

1. A login / authentication page before user of the app can get into the main content or page of the mobile app; the login can be done through the authentication token, or communicating with a remote server
2. A map for data visualization; in particular, they want to be able to pinpoint the map to get the location and visualize the GPS coordinates on the map

3. Several forms for describing person or place; these forms will be used for collecting data about people and places
4. A messaging, or push notification mechanism for communication; this is helpful for publishing or receiving information

Given these needs, I have developed four templates and each one corresponds to each of the functionalities. The following is the list and their descriptions:

1. Login/Authentication template
    a. Developer of the app can select local login or remote login
        i. Local login uses authentication token
        ii. Remote login uses server authentication with a username and password
    b. Users of the app can only advance to the next page if they pass through the the login page
2. Google Maps template
    a. People can pinpoint the location on a map and get back their GPS coordinates
    b. Developer of the app can visualize a collection of GPS coordinates on the map
3. Form templates
    a. Describe a person in Linked Data Form
    b. Describe a place in Linked Data Form
4. Google Cloud Messaging template
    a. Includes a copy of the remote server code which is essential for this template
        i. Developer can host the remote server code within their Google Drive
    b. Registers to a remote server and receive close to real-time or periodic push notifications from this server
    c. Requests to send messages to all the devices registered with the remote server
    d. Unregisters with the remote server and unsubscribe to all the push notifications

7.2 A Mobile App Development Case Study

These templates provide many basic elements for a good mobile application. I am going to present a mobile app which I find common in many of the occasions. I will walk through the steps on how to develop this mobile app using the Mobile Linked Data App Kit.

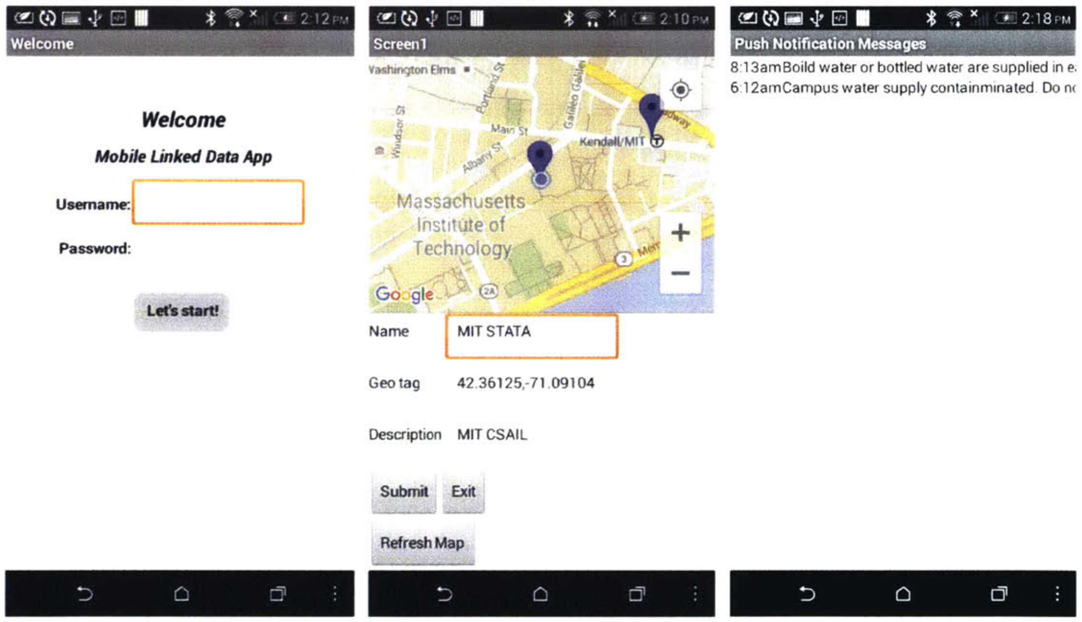
This mobile app needs to publish information about a particular location, such as the name of the building or the name of the business, the GPS coordinates, and a detailed description of the locations. Ideally, the users of the app should get the GPS coordinates by clicking on a map. In addition, the app needs to be able publish such information in Linked Data format to a triplestore. The app also needs to be able to fetch all the information from the triplestore and then visualize it on the map as pins. Users of the app can click on the pin and then see the information for that particular location. Users should have the option to refresh the maps since the data in the app is asynchronous with the triplestore. The apps should also be able to subscribe to a remote server and get messages or push notifications from there as well. Lastly, users need to provide their username and password in order to use this app.

A mobile app can be created without the Mobile Linked Data App Kit, but it can be time consuming. When using the Mobile Linked Data App Kit to develop an app, the developer of the app can just reuse the existing templates, load them into the project, start modifying the templates, and create a new project. Here is the list of the essential steps:

1. Load the authentication template into the system as the first screen
2. Supply the authentication template with a given remote server URL to verify the username and password
3. Add a simple logic in the block editor to advance page if the server has successfully verified the credentials
4. Load the Google Map template as the second screen
5. Generate the form for place in the second screen, which has the name field, GPS field, and description field.
6. Use the Docker Virtuoso container to bring up a Virtuoso instance for storing the Linked Data and take note of the server URL
7. Supply the form in step #4 with the updated server URL from step 6

8. Add a simple logic in the block editor - update the GPS field in the form whenever the user clicks on the map
9. Add a simple logic in the block editor - provide the Google Map component with the list of the data return from the triple Virtuoso data server when the user requests to update the pins on the Google Map
10. Load the Google Cloud Messaging template as the third screen
11. Supply the Google Cloud Messaging component with the given remote server, in order to receive the push notification.

These are the essential 11 steps for putting the app together. People can add more customizations to that as well, maybe for branding purposes, etc. Below are some screenshots of the finished mobile Linked Data app.



Name MIT STATA

Geo tag 42.36125,-71.09104

Description MIT CSAIL

8:13amBoild water or bottled water are supplied in e

6:12amCampus water supply containminated. Do n

Figure 7-1: (left) Login screen
(center) Google Map and form for place
(right) Push notification for messages

7.3 Summary

The Mobile Linked Data App Kit simplifies the process of building a mobile app. In particular, it makes generating and consuming Linked Data easier by using the Linked Data Extension during development of a mobile app. It also automates the data store deployment process by using the Docker Virtuoso container. The mobile app in the case study is an advanced-to-sophisticated one because it involves a remote server login, Google Map visualizations, and push notifications. However, such a complicated mobile application can be created in the Mobile Linked Data App Kit in just 11 steps.

7.4 Final Version of the App-Building Methodology

Here is the final version of the methodology that includes the updates from the toolkit and evaluation.

1. Model the software visually
    a. Mock-up the user interface on paper first
    b. Break the user interface into components
    c. Capture the structure and behavior of each component and write them down
2. Manage requirements
    a. Present the requirements to the colleagues in the team early
    b. Reach consensus with colleagues on what the system should and should not do
    c. Understand that user requirements evolve over time and adapt to change in needs
4. Use the Mobile Linked Data App Kit first
    a. Try to avoid writing new code
    b. Check if the features needed are covered in the collection of templates
        i. Use templates if they are available and matched the needs
        ii. Modify the templates if needed

   c. Use the Linked Data Form Generator to create new forms
   d. Use the Copy and Paste feature in the Linked Data Form Extension if you need to duplicate forms within the mobile app
5. Develop iteratively
   a. Back up the project on a regular basis
   b. Set and manage objective milestones
   c. Identify the minimum viable products and each additional features
   d. Have initial iteration for early feedback and do testing and integration as well
6. Verify quality
   a. Write test plan on each component to verify functionalities
   b. Invite people to test the app early based on the test plan
   c. Diagnose and fix bugs as soon as you discover it in the app

Chapter 8

Conclusion and Future work

This chapter summarizes the important contributions of this thesis and presents future research directions.

8.1 Thesis summary

Mobile devices are reshaping the field of disaster management. These relief workers are not experts in building mobile apps but they do have the need for specifically tailored apps that need to be built on a moment's notice. They understand the mobile apps' potential requirements but lack the ability to quickly implement them. There are many challenges associated with the current practices, such as a lengthy development cycle, costly budget, etc. The contribution of this thesis lies in exploring how to bridge this gap by identifying the crucial and needed requirements for humanitarian mobile applications, by ameliorating the obstacles encountered when building such mobile applications using the Punya framework, and by introducing such a framework to relief workers. This will allow relief workers to rapidly build humanitarian and disaster relief mobile applications. This thesis has made significant contributions to the current body of mobile development in the humanitarian domain in the following aspects.

Firstly, this thesis demonstrates the importance of having mobile Linked Data applications assisting the aid workers. This was achieved during the participatory design workshop and in interviews with the aid workers from the International Committee of the Red Cross. Participants from the participatory workshop were able to create apps from concepts they had developed using the Punya framework. Participants highlighted the need to have a template app that has reporting and subscribing features.

Secondly, this thesis demonstrated that humanitarian workers can develop and deploy mobile applications using the Punya framework given proper guidance. We also identified an area of improvement for the Punya framework to further enable aid workers to easily build humantain mobile applications.

Thirdly, this thesis introduces a Mobile Linked Data App Kit. The demonstration in Chapter 6 shows that the kit makes it easier for people to develop mobile applications which produce and consume Linked Data. The Virtuoso Docker container within the kit helps to create an instance of Linked Data store quickly and conveniently.

8.2 Message

The capability to quickly create an app and modify it rapidly is critical to crisis response management. Throughout the workshop, interviews, and experiment, we found that the feature to reuse and modify an existing app is greatly needed by our participants. The app-building framework allows its users to upload an existing app, modify the app, and create a new app. In addition, we discovered a similar experience in the Linked Data Generator, which enables users to use an existing form, modify that form, create a new form, and share that new form. This use-modify-create mechanism allows different entities to take apps created by others and collaboratively improve on or customize them to suit their immediate needs.

8.3 Future work

There are several directions to pursue for future work. First of all, the current Linked Data Form Extension doesn't support generating forms that involve multiple levels of ontologies directly. For example, a fire ontology might include the FOAF (Friend Of A Friend) ontology, Organization ontology, or Place ontology, etc. Second, it would be highly beneficial to have a gallery for form templates, or a feature which is designed to encourage sharing and exploration of forms. Because people who use the current Linked Data Form Extension need to build the fire ontology manually. People should be able easily publish their finished forms to the gallery, modify and create new forms, and rate and comment on them. Furthermore, it is not possible to merge two different forms at different screens at this point. This will be a nice additional feature to have as people need to consolidate the forms within the mobile app sometimes. This will requires copy and paste feature at the component level. Finally, we have shown based only on our observations that that the Linked Data Form Extension simplifies the form generating process. It would be ideal if we could support this claim using qualitative research or user studies.

Bibliography

[1] Anubhav Jain, Julius Adebayo, Eduardo deLeon, Weihua Li, Lalana Kagal, Patrick Meier, and Carlos Castillo. Mobile Application Development for Crisis Data. *Proceedings of Humanitarian Technology: Science, Systems and Global Impact (HumTech),* Pages 255–262. Procedia Engineering Volume 107, 2015.

[2] Tim Berners-Lee. Linked Data Design Issues. World Wide Web Consortium, July 2006. http://www.w3.org/DesignIssues/LinkedData.html.

[3] Grigori Babitski, Simon Bergweiler, Olaf Grebner, Daniel Oberle, Heiko Paulheim, and Florian Probst. SoKNOS - Using Semantic Technologies in Disaster Management Software. *The Semantic Web: Research and Applications*, V2:183-197, 2011.

[4] Kobotoolbox Platform. Harvard Humanitarian Initiative, March 2015. http://www.kobotoolbox.org/.

[5] Carsten KeBler and Chad Hendrix. The Humanitarian EXchange Language: Coordinating Disaster Response with Semantic Web Technologies: Coordinating Disaster Response with Semantic Web Technologies. *Semantic Web Journal* 6.1: 5-21, 2015.

[6] Sigmund Kluckner, Katrin Ellice Heintze, and Willi Wendt. Designing for the User: Tailoring a Simulation Software Interface to the Needs of Crisis Managers. *Proceedings of 11th International Conference on Information Systems for Crisis Response and Management*, 2014.

[7] Jerome David and Jerome Euzenat. Linked Data from Your Pocket: The Android RDFContentProvider. *Proceedings of the 9th International Semantic Web Conference*. Page 129-132, 2010.

[8] Weihua Li, Julius Adebayo, Fuming Shih, and Lalana Kagal. The Role of Mobile Technologies in Humanitarian Relief. *Proceedings of the 12th International Conference on Information Systems for Crisis Response and Management,* 2015.

[9] Minu Limbu. Management of a Crisis Vocabulary. Open Knowledge Foundation, July 2013. http://lov.okfn.org/dataset/lov/vocabs/moac&gt.

[10] Linked Data. World Wide Web Consortium, September 2014. http://www.w3.org/wiki/LinkedData.

[11] Mathieu D'Aquin, Fouad Zablith, and Enrico Motta. Wayou - Linked Data-Based Social Location Tracking in A Large, Distributed Organisation. 2010.

[12] MIT App Inventor About US. MIT Center for Mobile Learning - the App Inventor Team, December 2015. http://appinventor.mit.edu/explore/about-us.html.

[13] MIT Punya. MIT Computer Science and Artificial Intelligence Laboratory - Decentralized Information Group, December 2015. http://punya.mit.edu.

[14] Shyamal Mitra. Best Practices in Software Engineering. CS 312 - Introduction to Programming, University of Texas at Austin, August 2015. http://www.cs.utexas.edu/~mitra/csSummer2012/cs312/lectures/bestPractices.html.

[15] Vocus PR-Web. AppMakr Unlocks The Power Of The Socially Mobile Experience, Debuts Industry's First 'In-App' Social Network. The Street, March 2011.

http://www.thestreet.com/story/11042331/2/appmakr-unlocks-the-power-of-the-socially-mobile-experience-debuts-industrys-first-in-app-social-network.html.

[16] Jaziar Radianti, Julie Dugdale, Jose Gonzalez, and Ole-Christiffer Granmo. Smartphone Sensing Platform for Emergency Management. *Proceedings of 11th International Information Systems for Crisis Response and Management*, 2014.

[17] Faisal Razzak, Dario Bonino, and Fulvio Corno. Mobile interaction with smart environments through linked data. *International Conference on Systems Man and Cybernetics (SMC),* Page 2922-2929. IEEE, 2010.

[18] Francois Scharffe, Atemezing Ghislain, Raphael Troncy, Fabien Gandon, Serena Villata, Benedicte Bucher, Faycal Hamdi, Laurent Bihanic, Gabriel Kepeklian, Franck Cotton, Jerome Euzenat, Zhengjie Fan, Peierre-Yves Vandenbussche, and Bernard Vatant. Enabling linked-data publication with the datalift platform. *Proceedings of AAAI workshop on semantic cities*, 2012.

[19] Fuming Shih, Oshani Seneviratne, Daniela Miao, Ilaria Liccardi, Lalana Kagal, Evan Patton, Patrick Meier, and Carlos Castillo. Democratizing Mobile App Development for Disaster Management. *IJCAI Workshop on Semantic Cities,* 2013.

[20] SPARQL Protocol for RDF. World Wide Web Consortium, December 2015. http://www.w3.org/TR/rdf-sparql-protocol.

[21] Annemijn Van Gorp. Integration of Volunteer and Technical Communities into the Humanitarian Aid Sector: Barriers to Collaboration. *Proceedings of 11th International Information Systems for Crisis Response and Management*, page 622-31, 2014.

[22] Eric Von Hippel. Lead Users: A Source of Novel Product Concepts. *Management Science* 32 (7):791–806, 1986.

[23] Matthew Wall and Gabriella Mulligan. How Mobile Tech Is Improving Global Disaster Relief. Technology of Business - BBC News, December 2015. http://www.bbc.com/news/business-34715962.