


2012

A rapid deployment model for VGI projects in mobile field data collection

Suganya Baskaran
Iowa State University

Follow this and additional works at: <https://lib.dr.iastate.edu/etd>

 Part of the [Computer Sciences Commons](#), and the [Geographic Information Sciences Commons](#)

Recommended Citation

Baskaran, Suganya, "A rapid deployment model for VGI projects in mobile field data collection" (2012). *Graduate Theses and Dissertations*. 12793.
<https://lib.dr.iastate.edu/etd/12793>

This Thesis is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

A rapid deployment model for VGI projects in mobile field data collection

by

Suganya Baskaran

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Science

Program Of Study Committee:

Christopher J Seeger, Co-major Professor

Leslie L Miller, Co-major Professor

Simanta Mitra

Iowa State University

Ames, Iowa

2012

TABLE OF CONTENTS

LIST OF FIGURES	iv
LIST OF TABLES	v
ACKNOWLEDGEMENT	vi
ABSTRACT.....	vii
CHAPTER 1. INTRODUCTION	1
1.1 Background.....	1
1.2 Role of VGI in Community Extension	1
1.3 Need for Study	4
1.3.1 Storage and Consumption of Spatial Data.....	4
1.3.2 Challenges in Existing Systems.....	5
1.3.3 Research Objectives	7
CHAPTER 2. LITERATURE REVIEW.....	8
CHAPTER 3. MODEL.....	14
3.1 Technology Model.....	14
3.2 System Design	15
3.1.1 Smart Phone.....	15
3.1.2 Web Services	16
3.1.3 Web Authoring System	16
3.1.4 Geographic Information System.....	16
3.3 Software Framework.....	17
3.4 Assessment Creation.....	18
3.4.1 Spatial Meta Data Input.....	20
3.4.2. Questionnaire Input	21
3.4.3 Configuration Creation	23
3.5 GIS	23
3.5.1 Spatial Data base	23
3.5.2 Geoprocessing Module	24
3.6 Mobile Data Collection.....	26
3.6.1 iOS Cocoa Environment.....	26

3.6.2 Components of the Mobile Application	27
CHAPTER 4. IMPLEMENTATION AND RESULTS.....	32
4.1 Introduction.....	32
4.2 Case Study Description.....	33
4.3 Implementing an Assessment Creation.....	34
4.3.1 Databases.....	34
4.3.2 Implementing the Web Forms	35
4.3.3 Custom jQuery Rules for Validation	44
4.3.4 HTML Generation in jQuery	44
4.4 Implementation of GIS	45
4.4.1 ArcGIS Server	45
4.4.2 ArcSDE.....	45
4.4.3 ArcMap.....	46
4.4.4 ArcPy.....	46
4.4.5 Implementing the Geoprocessing Module.....	47
4.5 Implementation of Mobile Application.....	49
4.5.1 View Controllers in an iOS App.....	50
4.5.2 ArcGIS API for iOS	50
4.5.3 Implementation of Management Module	50
4.5.4 Implementation of Mapping Module.....	52
4.5.5 Implementation of Web Module.....	54
4.6 Evaluation Assessment	55
CHAPTER 5. CONCLUSION AND FUTURE WORK.....	58
5.1 Conclusion	58
5.1.1 Administrator Benefits	59
5.1.2 User Benefits	59
5.1.3 Cost Benefits	60
5.2 Future work.....	60
BIBLIOGRAPHY.....	62

LIST OF FIGURES

Figure 1. System Design.....	15
Figure 2. System Framework.....	18
Figure 3. Activity diagram for GIS metadata gathering	20
Figure 4. Cocoa in the architecture of iOS	26
Figure 5. Activity diagram for mobile data collection.....	28
Figure 6. Rapid application deployment model.....	32
Figure 7. PlayArea decision tree	33
Figure 8. Sidewalk decision tree.....	34
Figure 9. Initial form for assessment RecSurvey.....	36
Figure 10. Spatial metadata.....	37
Figure 11. Questions form - Initial	38
Figure 12. Questions form - Filled.....	39
Figure 13. Number of questions in each feature class	40
Figure 14. Part of XML for PlayArea feature class	42
Figure 15. Model builder	48
Figure 16. Published feature service.....	49
Figure 17. Mobile app - login & assessment	51
Figure 18. Mobile app - Project & Map editing mode.....	52
Figure 19. Mobile app - Layers & Geometry	53
Figure 20. Mobile app - Call out & Survey	54
Figure 21. Mobile app – Rating of an Assessment.....	56

LIST OF TABLES

Table 1. Data types and metadata required.....	38
Table 2. Mapping between questions form and XML representation	43

ACKNOWLEDGEMENT

I take pleasure in expressing my gratitude to Professor Christopher Seeger for providing me with this wonderful opportunity. Prof. Seeger has been a constant source of inspiration and motivation at work. I started this project with no prior experience in iOS programming. His patience and supportive nature has allowed me to learn this new skill which would later propel my career. This project has been a turning point in my life and I wholeheartedly thank Chris for believing in me. I would like to thank my co-major professor Dr. Leslie Miller for his detailed and constructive feedback. His attention to detail helped me improve my research work and strengthen my thesis. Dr. Miller's course on database systems helped me tremendously with this thesis work. His personal attention and motivating words were very encouraging and provided me with confidence. I would like to thank Dr. Simanta Mitra for agreeing to be my committee member. His valuable suggestions and thought provoking questions helped me shape my work to be more useful.

I would like to thank Iowa State University's Extension staff for their help and support. I would like to extend my gratitude to my parents, friends and God Almighty for being with me in all walks of life. My special thanks to Atma Mani for his unconditional support and motivation.

ABSTRACT

Spatial data collection in an organizational setup is growing in terms of the number of applications. While these applications are similar in providing the capability to collect and store spatial data, they are inherently different in the domains they cater to. For instance, they range from collecting data for facilitated Voluntary Geographic Information (VGI) like evaluating children's walkability to schools or visual quality assessment of a community to crowd sourcing or emergency dispatch and large-scale census gathering. Mobile devices are a great medium to collect such VGI. Several constraints with respect to time, manpower and efficiency contribute to the lack of a technology model to enable dynamic creation and delivery of such projects.

In this thesis, we address the research questions as to what a model for rapidly deploying a mobile VGI should look like and how to create one. We demonstrate this by creating a web-based project authoring system. The data collected from this system is fed into a Geographic Information System (GIS) model that automates creation of necessary spatial components and exposes them as Representational State Transfer (REST) services. An iOS mobile application consumes the web services and enables field data collection. The model also integrates multiple projects for a user while providing a domain specific means for collecting non-spatial attributes. Existing solutions for gathering data do not consider the relationship of attributes to one other. This thesis presents a dynamic decision tree implementation for the same, which improves efficiency and ensures correctness of the data collected in the field.

CHAPTER 1. INTRODUCTION

1.1 Background

Smart phones are increasingly becoming a significant part of people's lives. The number of mobile phone subscribers owning smartphones in the United States has risen to 59% in 2012 from only 41% in 2011(Nielsen, 2012). It is estimated that about 1 billion smart phone units will be shipped out in 2012 alone (Raento, Oulasvirta, & Eagle, 2009). More than 25% of the population use their smart phones for accessing internet instead of their computers (Pew Internet, 2011). In addition to the varied applications (apps henceforth) and the entertainment they provide, smart phones are spatially aware with inbuilt GPS receivers. This ability is being tapped in a variety of markets providing location-based services, mapping and routing utilities. Such popular apps include Google Maps, Yelp, GasBuddy, AccuWeather etc. Mobile smart phones are also capable of collecting data either implicitly in various tracking apps like in Nike+ running app or explicitly where they allow the users to collect data. The latter forms the basis of Voluntary Geographic Information (VGI), which is spatial data collected voluntarily by individuals, who can either be stakeholders, participants in a study, online users of a service or voluntary citizens.

1.2 Role of VGI in Community Extension

The field of VGI is a special case of the larger Web2.0 phenomenon known as user-generated content (Graham, 2010). VGI has the unique advantage of increasing public participation in a community, and helps policy makers to receive local data of that geographical area (Werts, Mikhailova, Post, & Sharp, 2012). There are several fields in which VGI plays a significant role. Evaluation of children's walkability to schools is an

example of participants or parents collecting the data. Community generated live data such as generation of real-time traffic updates through mobile apps can be of benefit to a person in a car a mile behind. This is a typical example of crowd sourcing (Kamel Boulos et al., 2011). An example of such a mobile app is Waze. Further, observations from eye witnesses have proved valuable in case of natural disasters like earth quakes (Poser & Dransch, 2010). In addition, VGI can also be used in retail market analysis in order to extract spatial distribution patterns of objects and analyze information on their associated attributes (Lwin, 2011).

Iowa State University's Extension and Outreach department has been very active in VGI projects. The department has offices in all counties in Iowa. It engages the public and takes help from volunteers in collecting spatial data related to social, economic or emergency preparedness related issues. The data is collected through various GIS tools and is used for the different types of applications mentioned above. It acts as a liaison between the community and the policy makers across the breadth of social and economic issues. Seeger (2008) discussed various such projects done in Iowa State University's Extension through facilitated VGI.

Some of the projects are listed below:

- River Note - A 70-mile river corridor study in 1999, that included an exhaustive inventory of views, historic sites, architecture and other points of interest collected by local citizens. An aerial photo was used as a background map, and participants used markers to create an entry.

- Digital Chip Game - A project to locate park elements such as parking lots, playgrounds, restrooms, boating etc. Multiple points could be mapped with comments. Both these projects used Macromedia Flash 4 and web enabled File Maker Pro Database.
- An asset-mapping project, which collected both spatial and non-spatial input from public to identify issues affecting the community. The online survey asked questions about current conditions, and how they can be improved.
- A Green Infrastructure Mapping project let participants map the green initiatives in their cities. Google Maps API (Application Programming Interface) was used to develop a custom data collection tool and included several GIS base layers.
- View shed Delineator - A highway corridor study that gathered information from a study group regarding various design options and site locations with respect to location and quality of views. This was done for more than 300 miles of roadway and 2000 square miles of adjacent land. Participants had the ability to toggle various GIS overlays and aerial images, zoom, pan and draw a view shed cone.
- iWalk - A study for developing safe walking and biking routes to kids to and from their schools by taking into account the barriers as well as neighborhood safety issues.
- A 3D Visualization project to evaluate proposed architectural and site layout changes. Spatially referenced 3D models were built using Trimble Sketchup, and participants could create, edit or query information.

1.3 Need for Study

Several tools exist for creating and sharing maps, like ArcGIS Online and Google Maps. There are also several spatial and non-spatial API's for various technologies to develop custom applications in web and mobile platforms. Some of them are Google Maps API for PHP and Javascript, ArcGIS API for Javascript, iOS and Android, Bing Maps AJAX control API etc.

1.3.1 Storage and Consumption of Spatial Data

A simple data collection project in which geometry is the only data collected can be hosted online on freely available map servers. This is one of the quickest ways to view the data collected. However in many cases, a dedicated GIS server is necessary for a variety of requirements. First, in order to perform spatio-temporal and network analyses on the geometry collected, geoprocessing and networking modules are to be created on the server side and published as web services. Two, non-spatial attributes associated with the geometry are to be collected. For instance, a school plotted on a map might need additional information like the name of the school, whether it is an elementary or high school, number of students in the school etc. Three, the confidentiality of the project demands that the data be stored within the organization.

A typical project creation on a GIS server is a multi-step process. It includes creating an empty feature layer inside the Spatial Database Engine (SDE), setting spatial reference and adding attributes to the feature layer, creating a map document and adding the feature layers thus built, assigning symbology to the layers, creating a Map Service Definition (MSD) for the map document, and finally publishing the map as a Feature

Service. Presently, these tasks are accomplished manually by use of various tools requiring heavy investment in man-hours.

The data can be consumed in a variety of ways. One way is to use a GIS viewer to display the spatial data on top of a variety of freely available base layers. Such maps are best used for web-based visualization and sharing. A more useful and interactive way is to develop web-based mashups using the open APIs listed above. The latter method facilitates analysis of data with user interaction for services like buffering, geocoding, finding routes with barriers and multiple stops, tracking moving objects, spatial interpolation, zonal statistics, etc.

1.3.2 Challenges in Existing Systems

An overview of some of the main challenges in the existing systems is briefly discussed in the remainder of this subsection.

1.3.2.1 Managing multiple projects. All of the projects in Iowa State's Extension and other VGI projects share a similar objective – collecting spatial data. Yet, the field in which they are used differ, e.g., one might be for collecting information on trees, and other might be for collecting information on bus-stops. Yet, separate developmental efforts are currently required in spite of their inherent common goal. There is a clear need to develop a framework which would integrate similar projects and at the same time distinguish them with respect to their application.

1.3.2.2 Lack of a unified framework for project creation. There are various steps involved in creating a VGI project and deploying it. The GIS part of creating and publishing services is done from a GIS Server as mentioned in Section 1.3.1. The field

data collection software consumes these services. When multiple projects are tied to one common framework, addition or removal of a data collection assessment does not require editing the mobile app's source code. This requires that an admin or a domain expert creating the assessment be granted privilege at various levels. Also, such a person might not have the required GIS or computer science knowledge. Hence project development and deployment need to be streamlined and automated in a centralized manner and the challenge resides in harnessing the right tools and techniques to do so.

Rapid deployment of new assessments is another key motivation to have such integrated development. In case of emergencies, accurate, timely and handy field data collection through crowd sourcing and VGI is vital (Lwin, 2011).

1.3.2.3 Human Computer Interaction (HCI) issues. Non-spatial attributes giving more information about the object are collected in most of the applications, however they are presented to the user all at once due to limitations in the mapping APIs. In most cases if the attributes are related to each other, presenting unnecessary questions to the user becomes not only redundant but also misleading. For e.g., if the answer to a question “Is parking available?” is “no”, there shouldn't be a follow-up question “How many parking lots are there?”. Participants might feel the need to answer such questions just because they are present. Self-administered surveys are known to have high error rates and inadvertently faking the data affects its accuracy (Patnaik, Brunskill, & Thies, 2009). This significant aspect is attributed to the difference in user behavior during field data collection. Such individual differences are very important in HCI and participants in field are known to have varied cognitive abilities (Whitney, Batinov, Miller, Nusser &

Ashenfelter, 2011). The technical design of the interface might reduce the usability of the software and hinder the participant's experience in providing the right data (Seeger, 2008). Thus it is hard to alter survey questions in e-form interfaces (Patnaik et al., 2009). A surprising statistic states that only 1% users of online communities contribute almost all of the data contributed (Nielsen, 2006). This reiterates the importance of having an intuitive and user-friendly system in place so that this small percent can be retained, if not increased.

1.3.3 Research Objectives

Based on the gaps in existing systems, and challenges described in Section 1.3.2, the following research objectives have been identified for this thesis work:

- To develop a framework that facilitates fast deployment of VGI assessments.
- To create an authoring tool for a new assessments. The tool is to be used by administrators or domain experts to provide information about the GIS metadata, information about the non-spatial attributes, and their hierarchy.
- To implement a toolbox for creating, manipulating and editing content such that it automates GIS resource management.
- To develop a decision tree implementation for non-spatial attributes thus providing a hierarchy based dynamic survey model.

CHAPTER 2. LITERATURE REVIEW

Mobile data collection, VGI, rapid deployment and the user behavior components involved in the thesis each have their own pool of literature.

Goodchild (2007) introduced the term VGI and defined it as voluntary collection, editing, describing or sharing of spatial data using the available web tools and techniques by individuals. Such user-generated content is collected using technologies and tools like Web2.0, georeferencing, geotagging and GPS. Some of the first developed VGI systems include Wikimapia, OpenStreetMap, and Google Earth. A study suggests that a vast majority of the early VGI initiatives bloomed shortly after Google released an API to its mapping services (Elwood, Goodchild, & Sui, n.d.). According to Butler (2006) these VGI systems ‘democratize GIS’.

One of the first and probably most popular crowd sourcing projects is Open Street Map (OpenStreetMap, n.d.) which attempts to create free online map of the world. Feature creation in the map is immediate and is committed as soon as the user saves. At the time of writing this thesis, Open Street Map reports 857,800 users, an increase from 200,000 in January 2010 (OpenStreetMap Stats, n.d.). A number of services have been created around Open Street Map. For instance, OpenStreetBugs (OpenStreetBugs, n.d.), is a tool designed to mark and correct detected errors in the datasets of Open Street Map. Disadvantages in such ventures include nonexistence of common feature catalogue and existing differences in level of details in geometries and attributes for different areas (Heipke, 2010).

Werts et al. (2012) created a website for public participation that let citizens upload photos and attributes with respect to soil and water conservation issues. It used ArcGIS Silverlight API to render map in the browser and provided capability to view the images in Google Picasa. It also had additional features like social media integration through RSS and Google Analytics. This is an example of facilitated VGI, as it is web based. It is however, sequential in its questionnaire and monolithic in its use case.

Spatial data collection is not immune to user and measurement errors, and it poses concerns regarding accuracy and loss of detail (Zhang & Goodchild, 2002). Seeger (2008) has deployed a variety of web based VGI projects in the fields of assets and inventory mapping, site programming, preference studies and design evaluation. They are described in Section 1.2. The information is acceptable only if the participant is local to the community the data is collected. This can be validated by a qualifier like the zip code or the number of years he or she has lived in the locality. Also, the more corroborative the information is collected, the more reliable the crowd sourcing or the crowd-wisdom becomes. OpenStreetMap Wiki (n.d.) echoes this by mentioning that if one person puts in inaccurate data, maliciously or accidentally, the other 99.9% of people can check it, fix it, or get rid of it. Quality control has also been proposed to be implemented by automated methods and peer reviews by specialists (Maué & Schade, 2008).

Cybertracker Conservation (n.d.) implemented one of the first mobile data collection tools, which used hand held devices in mid-1990s to collect animal behavior. It enabled non-literate animal trackers to tap a representative icon when a particular behavior was observed. It helped to obtain large quantities of geo-referenced data. Cyber-

tracker has since changed to include form design and web synchronization of data. However the system interaction and behavior remain constant.

CAM by Parikh, Javid, and Sasikumar (2006) is another tool for data collection in developing regions. It extends paper forms with image and async support. Users first fill out paper forms, and then take picture from the phone's camera to trigger data entry.

A very popular toolkit for building information services in developing regions was built by Hartung et al. (2010). Open Data Kit includes an Android based mobile application; it builds logic and includes backend server integration. The study was one of the first in building survey applications for a non-technical user. It is mainly used for surveying in developing countries with an option to collect positional data such as latitude and longitude. As this framework does not provide a map interface or include a GIS model, the scope of VGI projects that fit in it is currently narrow. It has however proved effective for application specific surveys and might be used for crowd sourcing non-spatial data collection.

Patnaik et al. (2009) evaluated the accuracy of data collection in mobile phones. Health workers were presented with a Java application with multiple-choice questions, and were asked symptoms-related questions. They were already trained on the interface and yet the error rate for e-forms came out to be 4.2% compared to 0.45% for voice-based surveys. The authors point out that the lack of flexibility in altering what information is collected is a limitation and it decreases the accuracy. In this thesis, we overcome this by having a dynamic questionnaire so that participants are not overwhelmed or confused with the format.

Zhong et al. (2010) used GPRS to collect field data in a form using .NET Compact Framework. The authors developed basic map functionality with pan and zoom modes. Only point geometry was collected and no non-spatial attributes were collected. However the main limitation of the work was the lack of two-way communication between the mobile interface and the GIS server.

Yan, Yu, Wu, and Ma (2009) present a similar study with additional features. The authors provide a more comprehensive implementation with destination searching and path planning using .NET Smart Device Framework. The work however stores geometry (points and lines), and non-spatial attributes in a text file and later require manual imports to GIS software for further visualization and processing.

(Lwin, 2011) offers client-server architecture for collecting field data in disaster zones using POP3 mail service. Software UM-FieldGIS was created and the geometry data was collected using a mobile phone or a separate handheld GPS device. Attribute data collected by the user had to be entered in a positional text format, for instance, a '/' is to be added between fields and a ',' between field values. These values are then sent to a predefined email address with a predefined subject and optional attachments. String substitution was used to convert short to full text.

As there is none or limited semantics in the data collection interface, users knew exactly what attributes they entered. Errors in accuracy and inadvertent mistakes in data entry are bound to happen, especially in an emergency situation. Lwin's study is more suited for a closed group with a specific need, like in the public facility mapping and less so for VGI where the participant base varies widely and in many cases, is random.

Kiwanja.net (n.d.) and RapidSMS (n.d.) are SMS based solutions to survey. While kiwanja.net (n.d.) collects unstructured data for short surveys, crisis reporting and election polling, RapidSMS (n.d.) collects data in a specific format. For instance, to send name and age, the syntax could be 'f:John l:Doe a:40;'. For longer surveys, multiple message exchanges are necessary. For crowd sourcing and VGI, this however might prove to be unreliable, expensive and eventually would discourage public from participating.

There are various form based solutions like Pendragon forms (Pendragon, n.d.-a, n.d.-b), Java 2 Platform Micro Edition (J2ME) based clients like EpiSurveyor (DataDyne, n.d.), CommCare (Dimagi Inc, n.d.) and JavaRosa (Atlassian, n.d.). These are from commercial vendors, yet lack appropriate digital signatures and users might be prompted with unclear dialogs (Hartung et al., 2010). However these are still a viable way to collect form data.

Harrison (n.d.) implemented an interactive decision tree, which is based on conditional branching for a fixed number of list elements. In this thesis, we have extended it to generate dynamic HTML controls and elements based on XML parsing.

Goodchild (2007) hinted that mobile devices are best suited for collecting VGI, however at the time of his paper, mentioned that the tools to exploit this potential did not exist. In this thesis, we have efficiently tapped the various relevant technologies available. We provide a scalable, user friendly and unified mobile VGI model from dynamically generated GIS datasets. This could be the first step in integrating Voluntary

Geographic Information in national authoritative database systems like the Spatial Database Infrastructure (Sabone, 2009; Seeger, 2008).

CHAPTER 3. MODEL

This chapter presents the system design and software framework. The components within the framework namely mobile data collection, GIS and assessment creation are described in detail.

3.1 Technology Model

A technology model helps to identify and build the necessary technical resources in order to meet the requirements and to answer the research questions (Microsoft, n.d.). Some of the questions could be “How would UI forms be rendered?”, ”What technology is used to access data from remote databases? ” and “How do the needs of the business drive the capabilities built in?”. It contains within it the development model, and therefore focuses on component development and reusing development resources including source code.

Such a technology model can be applied in a scenario where there are multiple GIS projects that involve field data collection. A project in the context of this thesis is a data collection exercise that is carried out in a particular geographical location. Users are granted access at project level. Assessment is a collection of projects. The model provides a single mobile application to manage the multiple assessments while preserving the context of each separate assessment. It also helps to manage multiple projects within each assessment.

3.2 System Design

The overall system proposed in this thesis is as shown in Figure 1. The system design includes smart phone, web services, a GIS and a new assessment creation web system. For example, in an emergency situation such as flooding, non-spatial attributes specific to the scenario like amount of flooding, number of houses damaged and number of people affected are used to create a configuration file, which is consumed by the web service through Hypertext Transfer Protocol (HTTP). Also geometries to be mapped like points, buildings, or roads are input to the GIS, which automates creation and deployment of necessary services. A single mobile application is used for data collection. It does not need to be updated for new assessments, and users can access the new assessments and projects immediately after deployment.

3.1.1 Smart Phone

A mobile application needs to be installed in a smart phone which would facilitate

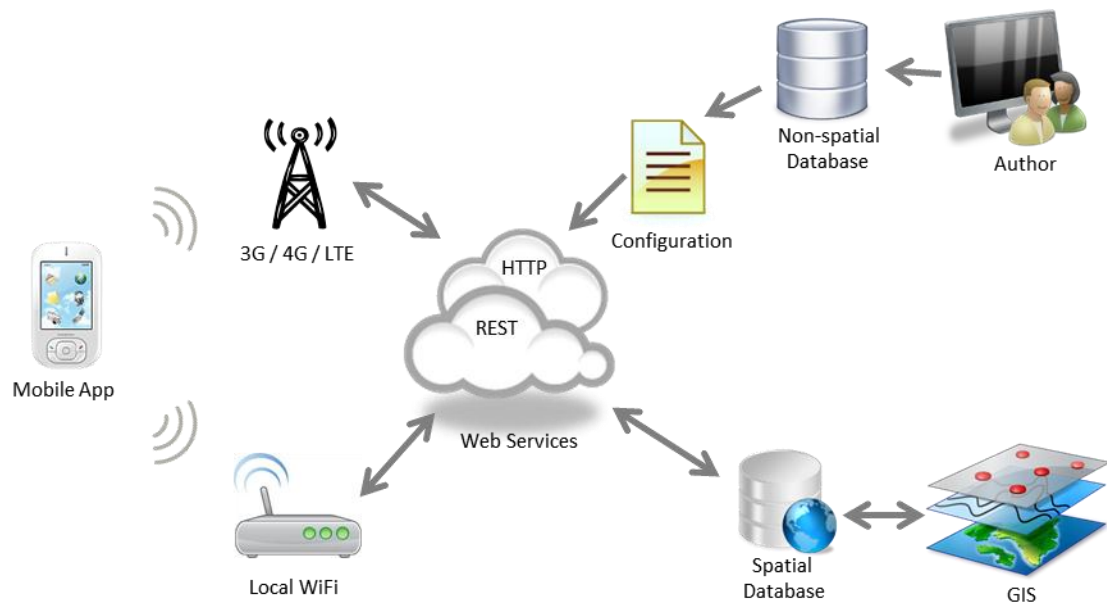


Figure 1. System Design

collection of geometries like point, line and polygon features and store them in the spatial data base engine of the GIS. It enables users to add new features, edit features they added earlier and view the features added by other users. The app uses Wi-Fi where a wireless access point is available; in the field it uses mobile data access through 3G or 4G or Long Term Evolution (LTE) capability provided by the service provider.

3.1.2 Web Services

Web Services enable interaction between the smart phone and other components in the model. REST web service is consumed by the mobile app to display the map and to add and edit spatial features. HTTP web service is used for user authentication, to retrieve assessments and to load the XML based dynamic web form to collect non-spatial attributes.

3.1.3 Web Authoring System

The Web Authoring System enables new data collection assessment creation. An administrator provides GIS metadata like the number of feature and map services and geometry of the features to be created. The administrator also enters the non-spatial attributes that need to be collected for the assessment, their relationship between one another and the HTML control in which they would be presented to the user.

3.1.4 Geographic Information System

A Geographic Information System helps to capture, store, manipulate, analyze and visualize geographic data (“GIS,” n.d.). The data is represented in a vector form, either as points, lines or polygons and is stored in spatial databases. One or more base

maps are used over which the feature layers are drawn. Adding, editing or removing the features is controlled through the mobile API.

3.3 Software Framework

The software framework for the thesis is as shown in Figure 2. The main components include the web assessment creation, GIS and the mobile smart phone. Each component has one or more modules. The components are loosely coupled. The framework provides flexibility in its implementation with a wide range of available software and techniques that is suited to the organization. Modules within each component generally interact with one other with high dependencies as they tend to fall within the scope of same programming language as is the case with this thesis. For example, the modules in the smart phone app are implemented through iOS cocoa framework and hence are programmed using Objective-C. Each of the components is explained in detail in the following sections.

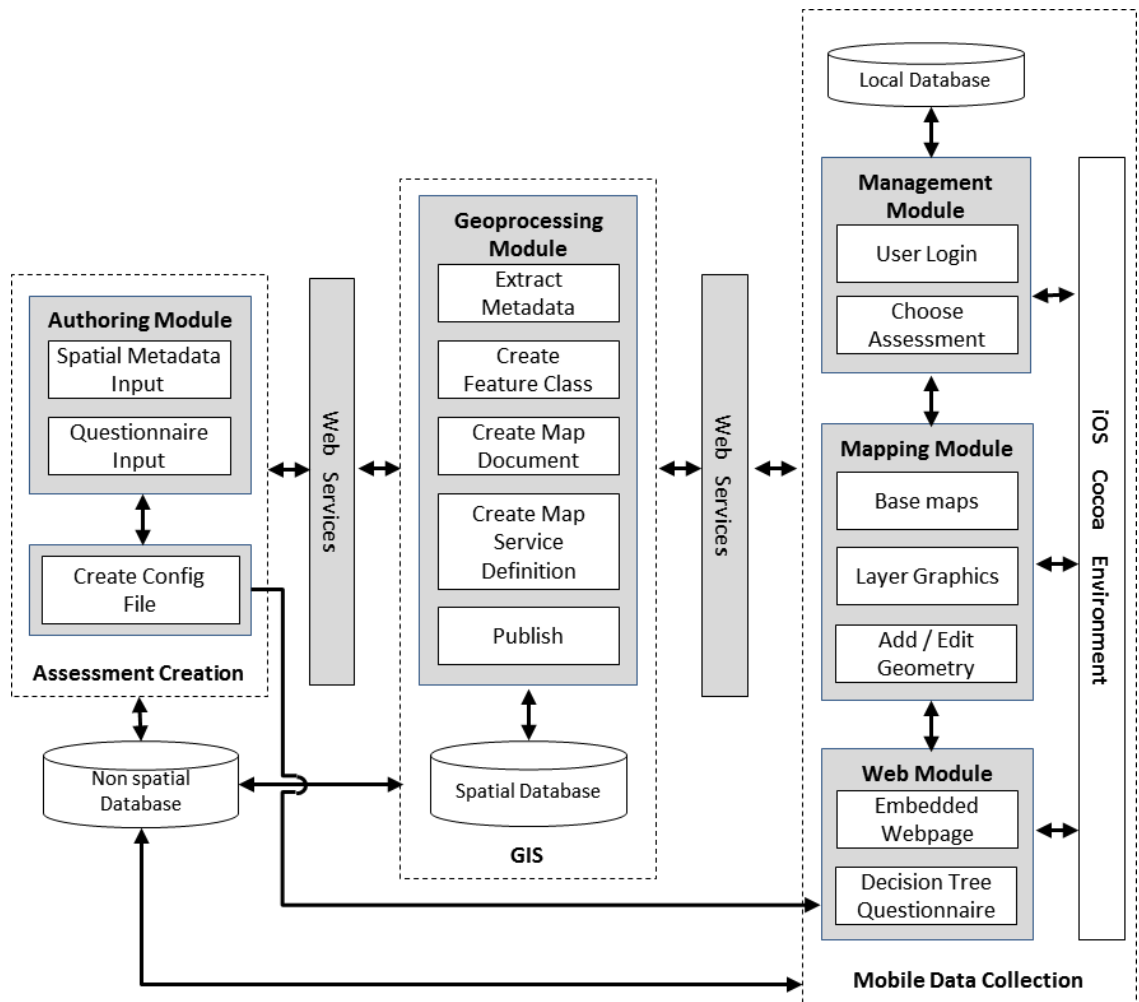


Figure 2. System Framework

3.4 Assessment Creation

An administrator creates the assessment. We have designed two types of assessments.

- **Data collection:** It involves users collecting various features in the mobile application. Users can create, edit or delete features, and answer questionnaires.
- **Evaluation:** It lets certain users evaluate data collected from other assessments. It could be used as a quality control measure. It can also be used to evaluate

services like restaurants, parks etc. In this case, the features already exist in read-only mode and the users rate the features based on a questionnaire.

Data collection is the primary assessment of this thesis while evaluation is an additional capability that was developed to extend the functionality of the system. Hence the vast part of this chapter focuses on data collection, while the evaluation part is discussed in Chapter 4.

In assessment creation, the first step in the bigger workflow, a variety of preliminary information related to the assessment is first gathered from an administrator. The authoring module involves collecting spatial metadata and questionnaire input. An activity diagram for GIS metadata gathering from admin is shown in Figure 3.

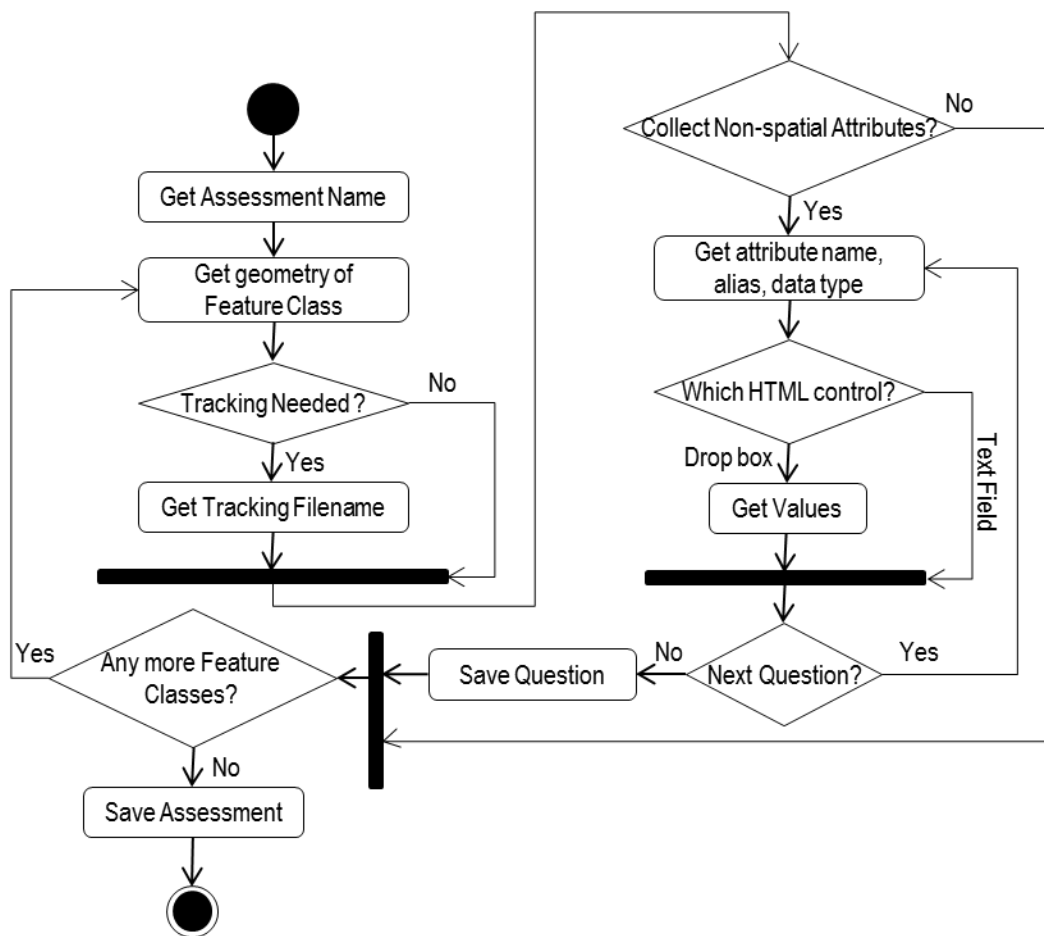


Figure 3. Activity diagram for GIS metadata gathering

3.4.1 Spatial Meta Data Input

A feature class is a GIS file that has the capability to store spatial features, their geometry, attributes and spatial reference. It can either be empty with no features or might have different features with symbology. A feature class when published becomes a feature service; features can be added, edited or deleted from a feature service. A map service is similar to a feature service except that the features in it can only be viewed and not edited. These files typically reside in a spatial database. They do not exist at the start

of the assessment creation and the administrator specifies the information needed to create them.

The tool to create assessments can either be stand-alone software or web based. The administrator provides details like the name and type of the assessment, number of feature classes and the geometry of each of those feature classes; she also supplies whether non-spatial attributes are to be collected for each of the feature classes. These non-spatial attributes represent the questionnaire part of assessment creation and are explained in Section 3.4.2. It is assumed that the administrator has the domain knowledge and has the questionnaire flow chart before the assessment creation. She can also enable tracking. In such cases, users can track themselves using point data at intervals as well as using a line route data. She also optionally provides map services that can be displayed on the map.

3.4.2 Questionnaire Input

Some types of features in an assessment require collection of non-spatial attributes as mentioned in Section 3.4.1. This section details how and what attributes are to be created. These attributes are fields in the feature class. In the front end, these are displayed as questions to the field user in the mobile; hence each attribute represents a question.

Since the number of attributes for each feature class differs, the addition of attributes or questions by the administrator is dynamic. The tool lets the administrator enter details for one question at a time and lets her add another question and so on.

For each of the questions or attributes, the admin enters the name of the attribute in compliance with the GIS software used and the alias of the attribute. This alias is exposed to the user in the front end as the question. For example, if the data type of the attribute is text, the permitted length is gathered. Similarly if the data type is double, precision and scale are gathered.

If the attribute takes predefined values, say 'yes' and 'no', the choice of answers for this question can be presented through a radio button, a check box or a dropdown menu. However if the user is required to enter a value, say 20 in response to a question 'How many parking lots are there?', a text field is presented to the user. Thus the admin also selects the HTML control in which the user input takes place. If the control is radio button or check box or drop down, the admin enters the predefined values that the attribute can take. Field data collection could take place under severe external conditions like snow or flooding and this design provides flexibility to the user in avoiding typing as much as possible.

Finally, one of the main disadvantages in using commercial off-the-shelf (COTS) software in presenting a questionnaire is the lack of dynamic nature. Users are presented with all the questions, whether relevant or not. For example, when the user picks an answer 'no' to the question 'Is parking available?' there should be no question 'How many parking lots are there?'. Thus the admin mentions what the next question should be for each of the predefined values for the current question. Once the information is collected for all attributes of all feature classes, the admin submits the form, which triggers the creation of the configuration file.

3.4.3 Configuration Creation

Part of information collected in Section 3.4.2 forms the input to this section. In order to dynamically load the questionnaire for the user, the questions are presented one at a time, and next question is based on the answer to the current question. The use of XML representation is helpful in such a situation and it prevents multiple calls to the database.

3.5 GIS

There are 3 types of GIS data that are collected, one is the meta data required to create the spatial files, second is the meta data required to create the non-spatial attributes. These two sets of information are used to create the feature class structure, and an attribute table. The third type of data collected is the actual field data by the mobile user. It comprises of the features in the features classes, and each feature represents a record in the attribute table.

3.5.1 Spatial Data base

A spatial database is a database that is optimized to store and query data in space like points, line and polygons (“Spatial Database,” n.d.). The Open Geospatial Consortium (OGC) sets the standards for adding spatial functionality to databases. Many spatial databases are readily available, both commercial and open source. For example, many well-known databases like Microsoft SQL, IBM DB2, Oracle, PostGreSQL support spatial capabilities and in some cases, they are integrated into an geographic information system so that the user can start using them like any other database. In addition to typical data types, a spatial database supports additional data types like geometry. It also supports spatial operations like spatial join, measurements, spatial predicates and other

spatial analysis functions. The feature classes are stored in one such spatial database so that querying and analysis can be done just as a regular database.

3.5.2 Geoprocessing Module

Geoprocessing is an operation to manipulate GIS data (“Geoprocessing,” n.d.). In this thesis, all GIS tasks are automated through the geo-processing module. It can either be through scripting or building a model or the combination of the two. A toolbox is created and the script or model is embedded in it. It is then published as a geoprocessing service. This service can be consumed externally as a web service.

This module is invoked after the configuration file creation described in Section 3.4.3. The following are the steps involved in the geoprocessing module:

3.5.2.1 Extract Metadata. Inputs to this task are the spatial metadata collected in Section 3.4.1 and the details of the non-spatial attributes in Section 3.4.2. This is done by a remote call to the external database. These extracted metadata form the input to the subsequent tasks.

3.5.2.2 Feature Class Creation. The feature class name and geometry submitted by the admin in assessment creation phase are used to create feature class. An additional spatial reference is also passed in. In the same way, each of the feature classes in the assessment are created and they are stored in the spatial database.

3.5.2.3 Map Document Creation. One or more feature classes form the input to a feature service. A map document is used to logically group these feature classes. The spatial reference of the first feature class added becomes the spatial reference of the map. The symbology for each feature class is also set. A feature class is represented as a layer with

a unique layer id in the feature service. For example, a feature service might have 3 layers, one for trees with point geometry, another for sidewalks with line geometry and another for buildings with polygon geometry.

3.5.2.4 Map Service Definition. The next step would be to scan the map for any errors or warnings, which might be suitability, visualization or performance related. An MSD file is then created which is an XML file. It is used to customize how the map is published; for example, it includes the data frame to be published and the settings for ‘anti-aliasing’, which is a method for creating better visualization for text and vector features. The MSD file is then published to the server.

3.5.2.5 Publish Feature Class. There are many ways a GIS task can be published to the server. It can then be consumed as a web service. For example, a geoprocessing service is used to run analytical models and automate tasks. A locator service is used in geocoding, that is, converting addresses to point geometry as well as reverse geocoding. Image services are used to generate integrated mosaic imagery to access raster datasets.

Map Services and feature services are of particular interest. They are both made up of one or more layers; the difference between them being map service is not editable. A map service definition is published as both map and feature service. The symbology and attributes of the features are also published. Once published, the services can be consumed, visualized and edited (in case of feature services) by many of the GIS visualizers online. They are also consumed using their service end points, typically Representational State Transfer (REST) URLs. In such a case, they are accessed through various web and mobile mapping APIs like ArcGIS API for Javascript, Google Maps

API, Bing AJAX API and so on. One such API is used in this thesis to consume the services in the mobile application for field data collection.

3.6 Mobile Data Collection

The mobile application in the smart phone is a significant part of the thesis. The implementation is currently done for iOS platforms.

3.6.1 iOS Cocoa Environment

The iOS Cocoa environment is the backbone for iOS development. Cocoa is the only application environment for iOS (“Cocoa Architecture,” n.d.). Cocoa has a runtime aspect, which presents the user interface components that are integrated with the underlying operating system. It also has a development aspect, which presents a vast suite of software building blocks that can be used as-is or extended. While there are few other languages to develop Cocoa software, Objective-C is the required and vital one and is used to develop the mobile app in this thesis.

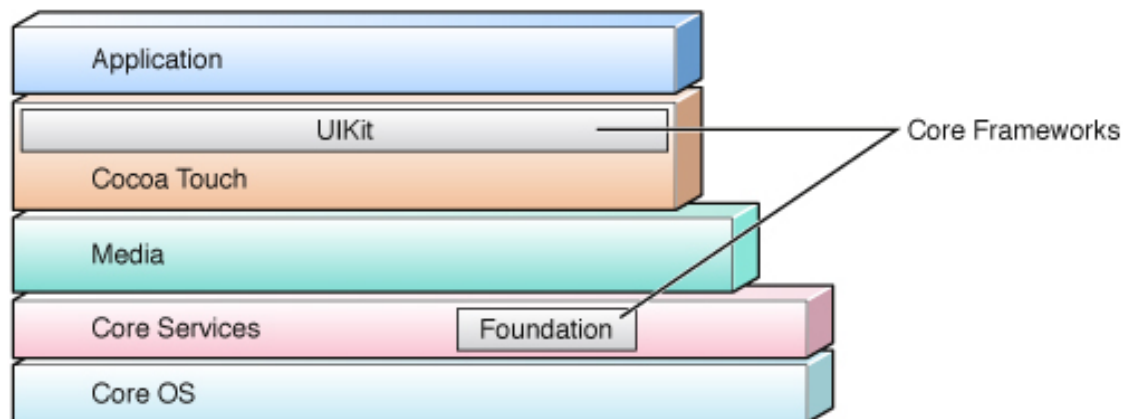


Figure 4. Cocoa in the architecture of iOS

Source: Cocoa Fundamentals Guide, Apple Inc.

Figure 4 shows how Cocoa fits into the architecture of iOS. It is a series of software layers from Core OS at the bottom to the application layer at the top. Some frameworks of each layer are listed below.

- Core OS: Includes kernel, file system, security and device drivers
- Core Services: URL utilities, string manipulation, preferences; hardware features like GPS, accelerometer, compass and gyroscope. The Foundation framework provides most of these services.
- Media: Graphics, OpenGL ES, animation, audio and video playback
- Cocoa Touch: Directly supports applications based on iOS user experience. The UIKit provides objects that are displayed and drawn in the UI including event handling.

3.6.2 Components of the Mobile Application

The components in this section include a local database and three modules namely management module, mapping module and web module. Figure 5 shows the activity diagram depicting the stepwise flow of the mobile application. The tasks performed by the various modules in the following sections.

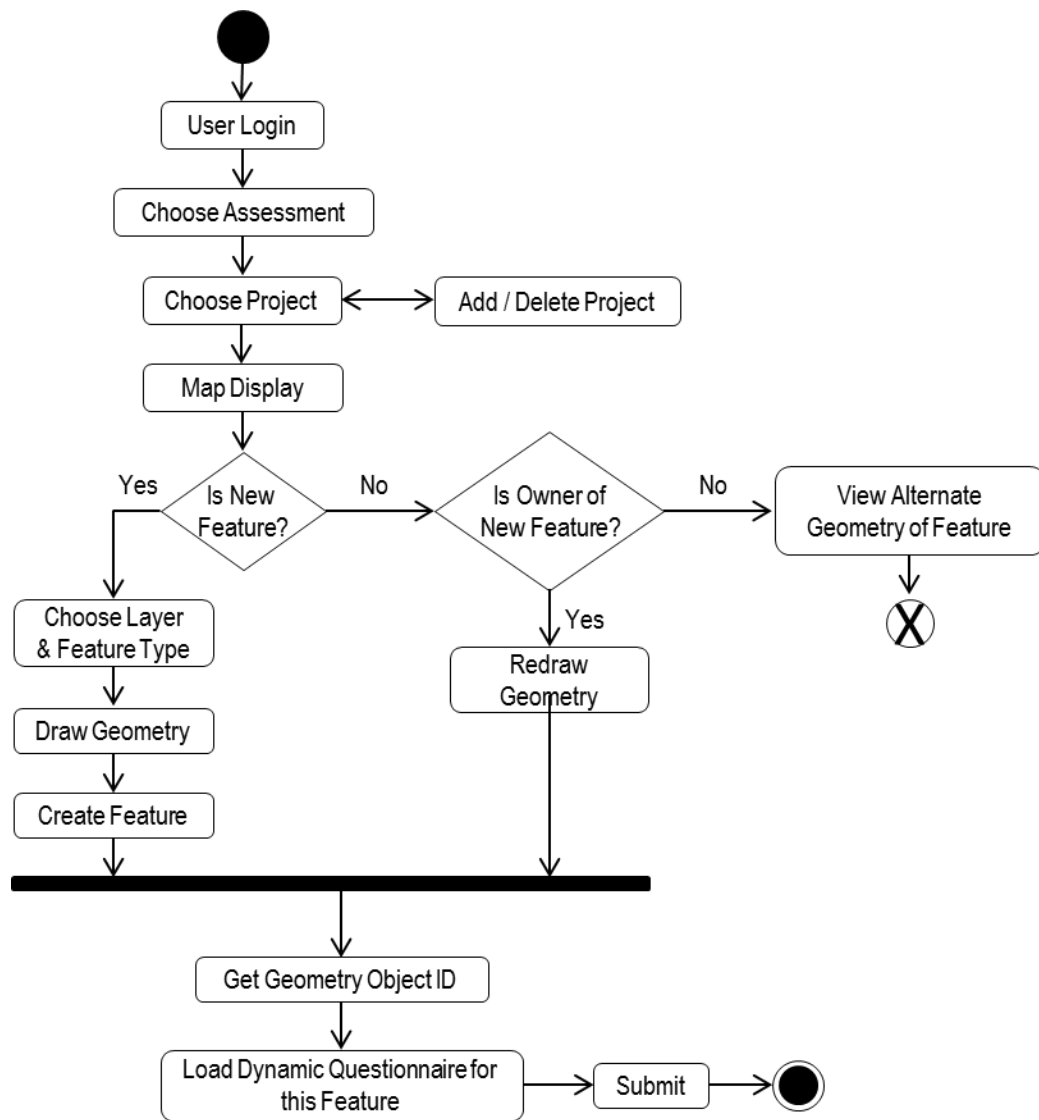


Figure 5. Activity diagram for mobile data collection

3.6.2.1 User Login. If the users are participants from a closed group, it is necessary to have user authentication. The authentication is not necessary when the project is spanned across to larger audience or public, as is the case when people in a community are visually assessing the quality of the town's infrastructure.

3.6.2.2 Assessment Choice. In case of closed user group, the same person might be taking part in multiple assessments, say one to map playgrounds in public schools and other to assess the quality of sidewalks in the town. This is significant as the thesis attempts to integrate multiple data collection assessments. When a new assessment is created by the admin, it's available to the user immediately.

An assessment and project might mean the same thing in a general sense. However in the context of this thesis, a project is specific to geography and an assessment is a group of similar projects. The app design is made this way because an assessment such as mapping the quality of sidewalks might be a statewide project while each town has its own set of local participants. Each town's data collection activity becomes a project. Thus users can add or remove projects they participate in.

3.6.2.3 Base maps. In any GIS system, base maps are necessary. These are the maps, which form the atlas over which the features are displayed. Examples of base map include street map, political boundaries map, world terrain map etc. The choice of the base map depends on the context of the application. Mapping of buildings might have a street map as the base map, as it needs the context of roads, intersections and streets; collecting rainfall or flooding data might use terrain map as base map. Base maps are accessed just like any other published service. While most of the base maps are available

online by commercial and open source vendors, they can also be built in-house if the project demands so. Base maps are visualized and consumed just like any other GIS service, either using an online or desktop visualizer, or by mapping APIs using the service end points.

3.6.2.4 Layer Graphics. This includes the feature services and map services. Both might have more than one layer with each layer having its own features and symbols. Field data collected is stored in feature service layers. The design of the system should permit the user to choose which layer they wish to collect data for feature services and permit layer selection and visibility for map services.

3.6.2.5 Add/Edit Geometry. The editable feature service contains graphics with geometry. The implementation should allow easy and flexible manipulation of the feature graphics and it should provide drawing tools to create new and edit existing features. It should also handle single and multi touch-based events based on user interaction. Users can edit the features they created earlier whereas they can only view the features created by others.

3.6.2.6 Embedded Web Page. Various mapping APIs studied for this thesis provided capabilities to retrieve all the attributes of a feature or a predefined fixed set of attributes. However this is not convenient in most situations in the field. The attributes are related to each other in most cases and the value of one might dictate whether another one is needed. In order to accomplish this, a web page is embedded into the mobile app and the attribute-entries are handled separately through a questionnaire.

3.6.2.7 Decision Tree Questionnaire. Once the user collects the spatial feature, control is passed on to the embedded web page for non-spatial attributes collection. This is done in the form of a questionnaire discussed in Section 3.4.3. The questionnaire is implemented in the form of a decision tree structure. It starts with one question and at any point, branching to next question depends on the answer to the current question. The unique object ID of the feature created just then is passed on to the questionnaire so that the system can associate the geometry of the feature and its non-spatial attributes.

3.6.2.8 Local Database. There are several pieces of information whose scope is the lifetime of the mobile app and in some cases extends the lifetime. For example, session maintenance might be necessary. Once a user is logged into the app for the first time, the app should remember her credentials and shouldn't ask her to log again. There are other data, which are just global variables whose values affect how the app behaves at any point of time. For example, the assessment the user chooses at the beginning of data collection determines what feature services are retrieved and where the data gets stored. This information is stored in a local database within the app and is available when the app is reopened. In most cases, the database is a file database.

CHAPTER 4. IMPLEMENTATION AND RESULTS

4.1 Introduction

The implementation of this thesis follows Rapid Application Development (RAD) model where the planning of the software development is interleaved with actual writing of the software itself (“RAD,” n.d.). It is shown in Figure 6 and has the following phases: a requirement planning phase in which the scope of the project is identified by the stakeholders, a user design phase in which users interact with system analysts to develop prototypes, construction phase in which project development takes place as in traditional software development lifecycle (SDLC), and a cutover phase which includes full-scale testing, migration and user training. The user design and construction phases are iterative as the system is built and involves continuous stakeholder participation.

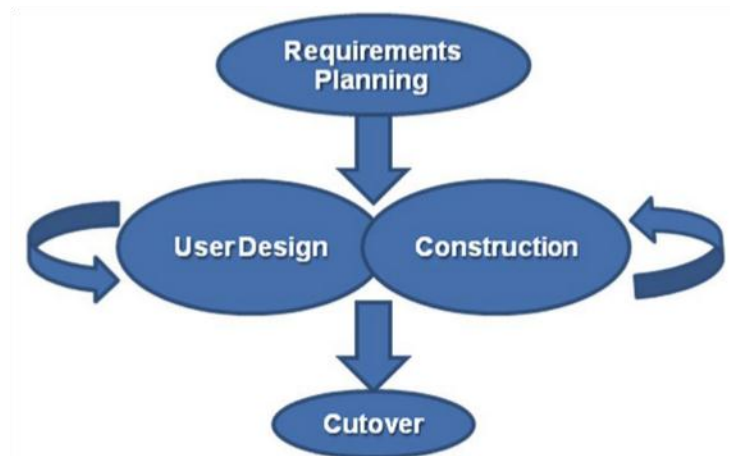


Figure 6. Rapid application deployment model

RAD model helps to build the system and get a working model faster than the traditional waterfall model. Hence the system can be deployed rapidly and more capabilities can be added to the system on a continuous basis.

4.2 Case Study Description

The case study describes an assessment for recreational survey that needs to be created and deployed. Let it be called *RecSurvey* and let it involve collecting two main types of data: One, play area features in a park, and two, condition of sidewalks en-route to the park. Spatial geometry is going to be point data, and the dynamic decision trees for the two types of data are as shown in Figure 7 and Figure 8.

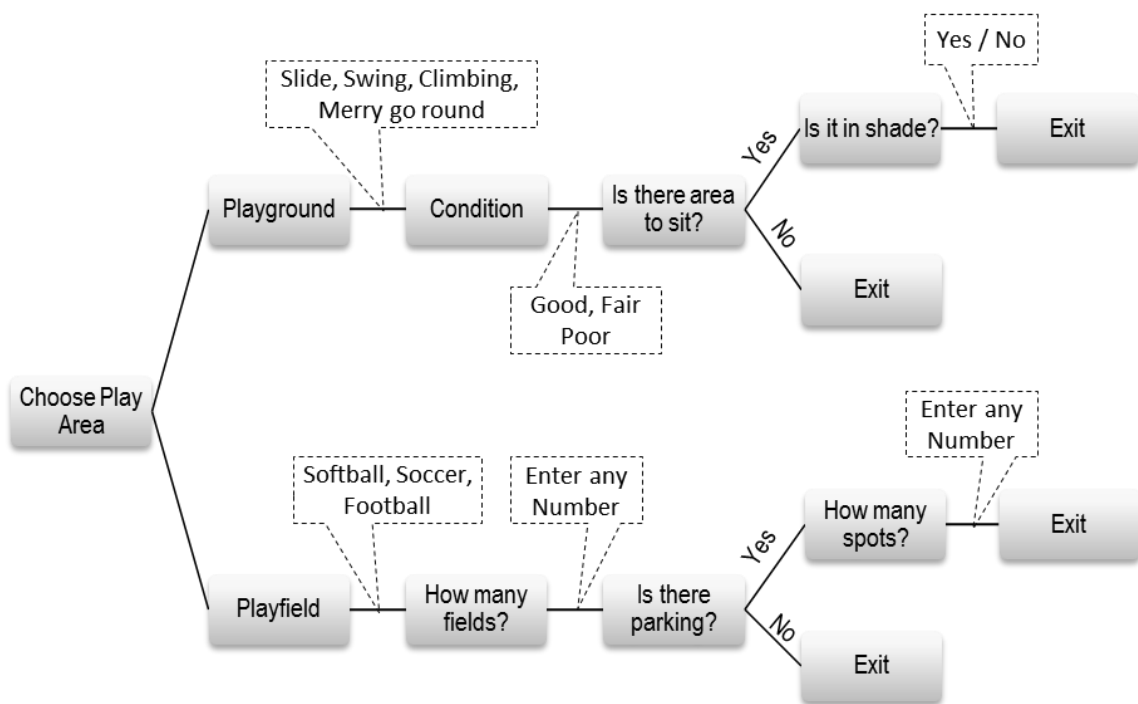


Figure 7. PlayArea decision tree

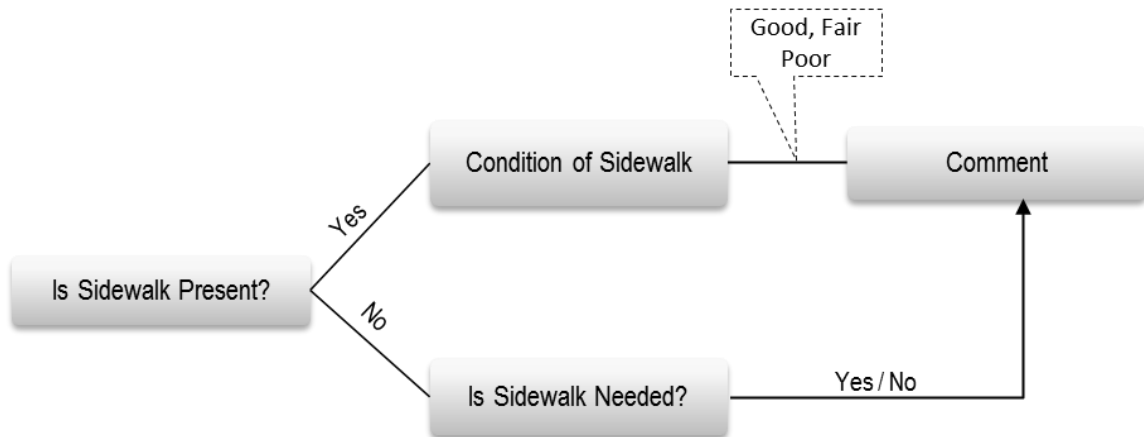


Figure 8. Sidewalk decision tree

4.3 Implementing an Assessment Creation

An administrator does the creation of assessment which is web based. It is assumed that the admin has the required domain knowledge. Forms are created and validated using technologies PHP, Javascript, AJAX and jQuery. Various plug-ins and modules of jQuery are extensively used including jQuery Validation Plug-in, jQuery UI Plug-in and jQuery Mobile.

4.3.1 Databases

Non spatial attributes are segregated from spatial attributes for ease of use and modularizing the design. There are various non-spatial databases available and MySQL is used in for assessment creation in the thesis. Navicat premium, which is one among the many GUI tools, is used to manage the database systems. The following gives brief explanation of the tables used:

- Assessment table has the generic information common to an assessment like the icon URL, number of feature classes in the assessment and so on.

- Feature class table has a record for each feature class in every assessment. The assessment ID is a foreign key in this table.
- Map Service table has entries of already existing map services that the admin wants to use in the assessment.
- Questions table has a record for each question for all feature classes and all assessments. Hence it is a master table for all questions. A unique Question ID identifies it. Feature class ID and assessment ID are foreign keys in this table.
- User info table has user details including their name, email and the projects they have access to.
- Projects are geography specific subsets of an assessment and the project table has details like location, city and the state the project is carried out.

4.3.2 Implementing the Web Forms

The assessment creation begins with an option to create two types of assessments: a data collection assessment or an evaluation. Figure 9 shows the initial form for the case study assessment: *RecSurvey. Evaluation* and rating of the data collected is an additional capability that is developed and is explained in Section 4.6. Hence *Data Collection* is chosen and the form is filled and validated for mandatory fields. Customized rules are created for complex validations and are explained in Section 4.3.3.

The screenshot shows a web browser window with the title 'Welcome to new Assessment'. The page content is divided into two sections:

- Choose Assessment Type:** This section contains two radio buttons. The first, labeled 'Data Collection', is selected (indicated by a blue dot). The second, labeled 'Evaluation', is unselected.
- Enter Assessment Details:** This section contains three input fields:
 - 'Assessment Name:' with a text input field containing the value 'RecSurvey'.
 - 'Choose number of Feature Classes:' with a dropdown menu showing the value '2'.
 - 'Number of Read Only Map Services:' with a dropdown menu showing the value '0'.

At the bottom left of the form, there is a 'Next' button.

Figure 9. Initial form for assessment RecSurvey

The values submitted by this form are captured in the next by PHP, which makes an entry to the assessment table. HTML contents are created dynamically based on the number of feature class and map services selected earlier. Creation of such HTML contents using jQuery is explained in Section 4.3.4. As the number of feature services selected earlier was two, there are two entries in the table as shown in Figure 10.

As mentioned in the case study, the feature classes are named 'PlayAreaFC' and 'SidewalkFC', and both are point data. If tracking were enabled, the system would create two more feature classes in addition, for user position tracking as points, and user tracks as line geometry. Tracking has not been implemented in the thesis and is in scope for the next iteration for the project. Feature class name is validated using custom regex rule and

map service text field is validated to contain URL. Then the contents are updated in feature class and map service tables in MySQL.

Sorting through the rows is done using jQuery UI which is built on top of the jQuery Javascript Library. It provides components for user interaction like dragging, sorting, selecting and so on. It also provides effects, widgets and themes (“jQuery UI,” n.d.).

The screenshot shows a web browser window with the title 'Feature and Map Service Det: x'. The main content area is titled 'Enter ArcGIS Feature Class Name and Geometry'. It contains a table with the following data:

Name	Geometry	Create Form
PlayAreaFC	Point	<input checked="" type="checkbox"/>
SidewalkFC	Point	<input checked="" type="checkbox"/>

Below the table, there is a checkbox labeled 'Do you want to enable tracking?' which is currently unchecked. At the bottom left, there is a 'Next' button.

Figure 10. Spatial metadata

Once the details are entered, they are updated to the database. A layout designed using jQuery UI list is displayed in a grid as shown in Figure 13. It gives a layout to choose a feature class to start adding questions.

On choosing PlayAreaFC, the questions form as shown in Figure 11 is displayed. It has one row initially. The question ID represented as ‘Global QID’ is unique to a question in all assessments. The delete button in each record and the ‘Add Row’ button at

the bottom can be used to delete and add questions respectively through AJAX calls to the database.

The screenshot shows a web browser window with the title 'Survey Questions'. The page content includes a header 'Welcome to Creating Survey Questions' and a sub-header 'Enter details for feature Class'. Below this is a table with the following columns: 'Del', 'Global QID', 'ArcGIS Attribute', 'ArcGIS Attribute Label', 'ArcGIS Data Type', 'HTML Control', 'Values', and 'Next Questions'. The first row of the table contains the following data: 'Del' (with a small 'Del' button), '62', an empty text field, an empty text field, 'Select One' (dropdown), 'Select One' (dropdown), an empty text field, and an empty text field. Below the table are two buttons: 'Add Row' and 'Save!'.

Figure 11. Questions form - Initial

In this form like the earlier ones, the whole HTML content of the form is created using jQuery and is explained in Section 4.3.4. The admin enters the attribute name, attribute label that is the alias and selects a data type of the attribute. Table 1 lists the data types and metadata required for each of the data types.

Table 1. Data types and metadata required

Data type	Meta data Required
Text	Length
Short Int, Long Int	Precision
Float, Double	Precision, Scale

The admin then chooses a HTML control in which the attribute would be collected from the user. Dropdown lists and text fields are implemented in this thesis. If dropdown is chosen, the values are predefined and entered in the *Values* column as comma separated entries. In this case, the *Next Questions* field takes Global QID entries

Del	Global QID	ArcGIS Attribute	ArcGIS Attribute Label	ArcGIS Data Type	HTML Control	Values	Next Questions
Del	62	playArea	Choose PlayArea:	Text 50	Drop Down List	Playground,PlayField	63,67
Del	63	playground	Playground Type:	Text 100	Drop Down List	Slide,Swing,Climbing	64,64,64,64
Del	64	pgCond	Choose Playground C	Text 50	Drop Down List	Good,Fair,Poor	65,65,65
Del	65	pgIsSit	Is there area to sit?	Text 50	Drop Down List	Yes,No	66,0
Del	66	pgSitShade	Is seating in shade?	Text 50	Drop Down List	Yes,No	0,0
Del	67	playfield	Playfield Type:	Text 100	Drop Down List	Baseball,Soccer,Football	68,68,68
Del	68	pfCount	How many playfields	Short Int 3	Text Field		69
Del	69	pfIsPark	Is there parking?	Text 50	Drop Down List	Yes,No	70,0
Del	70	pfPCount	How many parking Ic	Long Int 4	Text Field		0

Add Row

Save!

Figure 12. Questions form - Filled

also comma separated one for each of the *Values*. For example, if *Values* has ‘Yes, No’, a valid entry to Next Questions field is ‘100,101’ where 100 and 101 are QIDs present in this form. When the user selects ‘Yes’, the next question asked is the 100th. Similarly if the user selects ‘No’, the next Question asked is 101st.

If text field were chosen as the HTML control, the user in the field would enter the value. For eg, it would be an answer to a question like ‘How many parking lots are there?’. Hence the *Values* field may be left empty. In this case, the *Next Question* field contains just one QID. At any point of time, a QID of 0 represents end of decision tree. A filled out questions form according to Figure 7 looks like Figure 12.

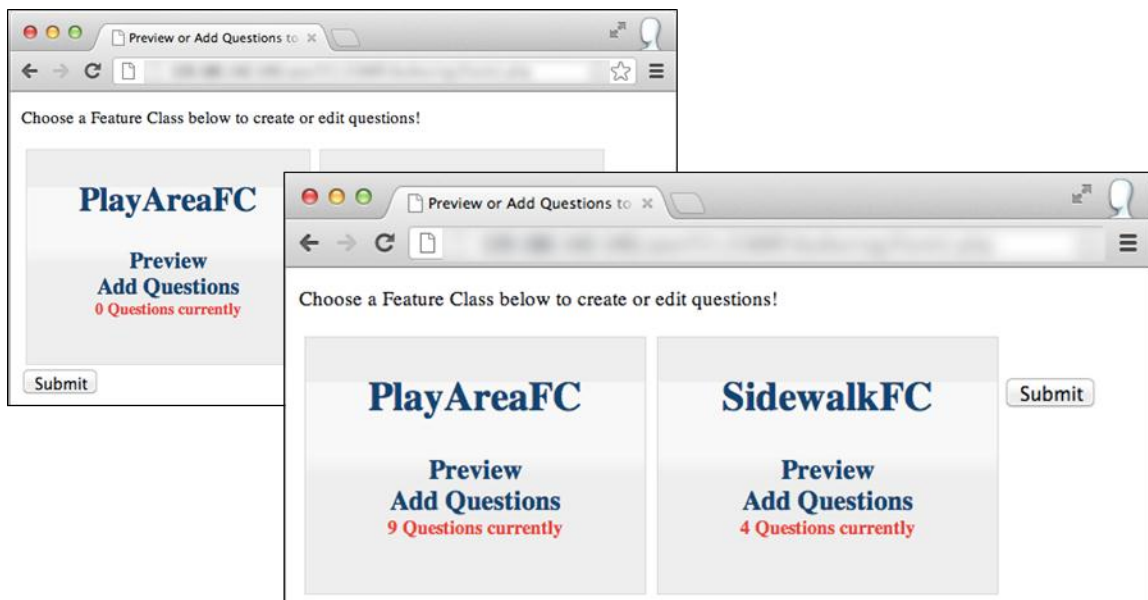


Figure 13. Number of questions in each feature class

Upon validation and saving, all questions are saved in the questions table using AJAX, and the previous form is loaded again. The number of questions for PlayAreaFC is now updated. It can be edited again by clicking. All the previously entered questions are loaded from the database. The questions can also be previewed in a read-only mode. In a similar way, SidewalkFC is filled as per Figure 8. The discussion will focus on PlayAreaFC. SidewalkFC is presented just to demonstrate multiple feature class handling. Initial number of questions and number of questions added to each feature class are as shown in Figure 13.

Upon the final submission, two important actions take place before the creation of assessment can be deemed complete: creation of the XML configuration file creation and triggering of a geoprocessing service.

4.3.2.1 XML Configuration File. Currently, each of the feature classes has different set of questions. In order to use them effortlessly without multiple database calls, the data needs to be represented in a more meaningful way. XML representation is used in the thesis to create the configuration file. It is then parsed and the questions are presented to the user.

PHP's XMLWriter extension of libxml xmlWriter API is used to create an XML file for each of the feature class and it represents the questions of that feature class. The files are stored at a specific location in the server from where they are parsed and displayed in mobile app. The files are named in the format tree_105_56.xml where 105 is the assessment ID and 56 is the feature class ID. This would ensure the app loads the correct questionnaire XML file depending on the mobile user's choice of assessment and project. Part of XML created for PlayAreaFC is shown in Figure 14.

```

▼<branch id="62">
  <content>Choose PlayArea:</content>
  <arcgisField>playArea</arcgisField>
  <inputcontrol>dropdown</inputcontrol>
  <metadataInfo>-</metadataInfo>
  <fork target="-">---</fork>
  <fork target="63">Playground</fork>
  <fork target="67">PlayField</fork>
</branch>
▼<branch id="63">
  <content>Playground Type:</content>
  <arcgisField>playground</arcgisField>
  <inputcontrol>dropdown</inputcontrol>
  <metadataInfo>-</metadataInfo>
  <fork target="-">---</fork>
  <fork target="64">Slide</fork>
  <fork target="64">Swing</fork>
  <fork target="64">ClimbingDevice</fork>
  <fork target="64">MerryGoRound</fork>
</branch>
▼<branch id="69">
  <content>Is there parking?</content>
  <arcgisField>pfIsPark</arcgisField>
  <inputcontrol>dropdown</inputcontrol>
  <metadataInfo>-</metadataInfo>
  <fork target="-">---</fork>
  <fork target="70">Yes</fork>
  <fork target="0">No</fork>
</branch>
▼<branch id="70">
  <content>How many parking lots?</content>
  <arcgisField>pfPCount</arcgisField>
  <inputcontrol>textfield</inputcontrol>
  <metadataInfo>0,4,0</metadataInfo>
  <fork target="0">---</fork>
</branch>

```

Figure 14. Part of XML for PlayArea feature class

The questions are displayed one at a time. The first question that is to be displayed is the one with the least *branch id*. This is the *QID* in Figure 12. The mapping between the values collected in Figure 12 and the XML file is shown in Table 2. *Content* is the actual question asked to the user. *Inputcontrol* shows how the user can submit answer for *Content*, whether selecting a value from a dropdown list or by entering a value in a text field. If *inputcontrol* is textfield, the answer the user enters has to be validated. The validation is based on the *metadataInfo* which is a comma separated values of length, precision and scale in that order. A positive value means that the value is present. For example, 20,0,0 means length is present but neither precision nor scale is present. Hence the value to be validated is a string and it can have maximum length of 20. Similarly 0,4,0 means integer with precision 4 and 0,4,3 means double value with precision 4 and scale 3. These are used for validation. If *inputcontrol* is dropdown, each *fork* tag has one possible value in the drop down menu. The value of attribute *target* of each *fork* tag

represents what the next question should be if this *fork* value is selected in the dropdown list.

Table 2. Mapping between questions form and XML representation

XML Representation	Questions Form Representation
Branch Attribute: id	Global QID
Content	ArcGIS Attribute Label
ArcgisField	ArcGIS Attribute
InputControl	HTML Control
MetaDataInfo	Length, Precision, Double
Fork	Values (parsed)
Fork Attribute: target	Next Questions (parsed)

The parsing of the XML is done by creating a dynamic decision tree builder. The framework for creating a decision tree with static questions and dropdown values alone was extended greatly to include any number of questions. Validation was also included and all HTML was created programmatically using jQuery instead of static HTML.

4.3.2.2 Triggering the Geoprocessing Service. ArcGIS API for Javascript is used to create a web component invokes a Geoprocessing Service (GP service) hosted on ArcGIS Server. The service is accessed using its REST end points. AssessmentID is the input parameter of the service. The service runs a geoprocessing tool box in the server which creates the actual PlayAreaFC and SidewalkFC feature classes in the server, assigns their geometry as points, creates attributes with correct data types specified by the admin. Implementation of the tool box in server side and publishing is explained in Section 4.4.

4.3.3 Custom jQuery Rules for Validation

jQuery validation plug-in's in built functions were used for simple rules such as for fields that are mandatory or those that can only be numerals. Validation is especially critical in the questions form in Figure 12 as the data type creation dictated in the form should comply with the GIS database's data types. For such complex validations, custom rules were created as part of the thesis. Some of the rules created were to validate

- If a regular expression in a text field must be alphanumeric.
- If *HTML Control* field is 'Dropdown', *Values* field must comply with ArcGIS data type and the meta data. *Next Questions* field should have the same number of entries as *Values* field and it can have only QIDs and commas.
- If *HTML Control* field is 'Textfield', *Values* field should be empty. *Next Questions* field in this case must have only one number, which is a QID.
- If *ArcGIS Datatype* field is float or double, the scale value entered must be less than the precision. Though this appears to be a simple conditional rule, it should be noted that there are multiple rows, and validation has to take place for each row. This is done by grouping the elements using jQuery classes and applying the rule for each member of the class.
- If a *HTML control* is selected before entering *Values* field.
- If an *ArcGIS Datatype* field is selected before entering *Values* field.

4.3.4 HTML Generation in jQuery

All of the web components expect the form itself, which acts as a frame to hold the rest of the HTML objects, are created dynamically using jQuery. While jQuery can

understand a HTML string and create one, it also has specific methods to create objects and attributes. This can then be attached to a table cell, which in turn can be attached to a table row which in turn to the table which in turn to the form itself. This makes the coding reusable with iterative calls to a function whenever a HTML object needs to be created.

4.4 Implementation of GIS

The GIS implementation is the link that ties the assessment creation with the mobile app. GIS services are created taking inputs from the former while the services are consumed in the latter. Software used in the implementation are ArcGIS server, ArcSDE with Microsoft SQL server, Arcmap and ArcPy.

4.4.1 ArcGIS Server

ArcGIS server helps to publish and share GIS resources like maps, address locators, geodatabases and tools within an enterprise or across the web. It differs from a normal server in that it also allows users to interact with the GIS functionality of the resource (“ArcGIS Server,” n.d.). For example, users can get directions to a point of interest from their current location using address locator functionality. An ArcGIS server can be deployed in an organization’s service oriented architecture (SOA) or remotely in a cloud environment.

4.4.2 ArcSDE

ArcSDE is the Spatial Database Engine for ArcGIS desktop and server. It is an application server and facilitates the usage of Relational Database Management Systems (RDBMS) for spatial data – both raster and vector. Currently, spatial data can be

managed using ArcSDE using one of the following databases: IBM DB2, Informix, Microsoft SQL Server, Oracle, and PostgreSQL (“ArcSDE,” n.d.). Since most of the existing projects in Iowa State University’s extension are hosted in ArcSDE using MSSQL in an ArcGIS Server, the same is used for the thesis.

4.4.3 ArcMap

ArcMap is the desktop application software used in ArcGIS. It is the central tool to display, explore, create or edit GIS data sets, assign symbols, create map layouts for publishing, use geoprocessing to automate work and perform analysis, organize and manage geodatabases and ArcGIS documents, publish map documents as map services using ArcGIS Server, document geographic information and customize the user experience. ArcMap also has several in-built toolboxes like 3D analyst tools, cartography tools, editing, geocoding and network analysis toolboxes. Each of the tools has a set of tools for a specific function. For the purposes of this thesis, we create a custom geoprocessing toolbox.

4.4.4 ArcPy

ArcPy is a Python package that provides access to geoprocessing tools as well as to additional functions, classes, and modules that facilitates creation and automation of workflows in ArcGIS (“ArcPy,” n.d.). It helps us to perform geographic data analysis, data conversion, data management, and map automation with Python.

4.4.5 Implementing the Geoprocessing Module

The output of this module is a geoprocessing service. This module is created once, and for every assessment, the task is run from a web interface and necessary GIS services are created and published.

A custom toolbox and a model within it are created. The model is created using Model Builder, an application in which we create, edit, and manage models. Any variable in the model that is exposed outside has to be designated as a model parameter. Hence the input, which would be passed as parameters from the web service consuming the GP Service, and the output that would be passed indicating success or failure, are designated as model parameters. Between the input and output components, several existing tools like creating a buffer, performing a spatial intersect etc. can be inserted. In this thesis, as the ArcPy script for creating services is included between the model parameters. The model created looks as shown in Figure 15. All of the automation tasks are run in the *Assessment Run* script.

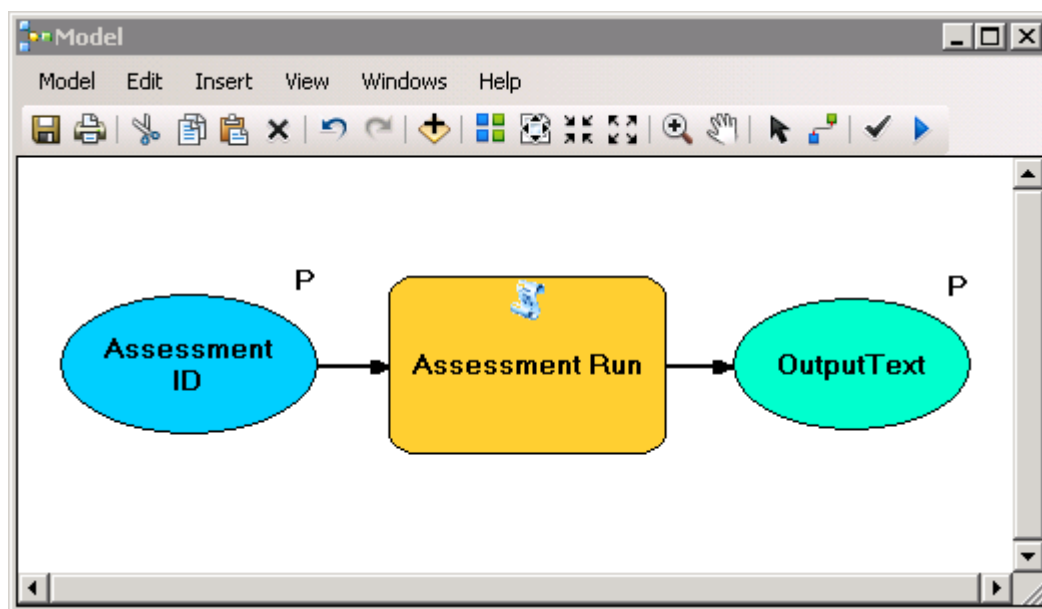


Figure 15. Model builder

The external python file to execute is added to the script. Also added are the input and output parameters and their data types. GIS files like feature classes can also be passed as parameters.

4.4.5.1 Scripting in ArcPy. ArcPy scripts read the metadata from MySQL server. A feature class is created using the name and geometry. Then for each non-spatial attribute entered by the admin during assessment creation, a field is added to the attribute table of feature class. Data type of the field and its associated length, precision or scale is defined accordingly. The feature classes are created in ArcSDE instead of a static physical file location. This also enables concurrent edits from the users. All feature classes in an assessment are grouped into a feature set in the SDE.

The script then creates a map document and symbology is assigned by the admin. The script also checks the map document for errors and warnings, creates a map service

definition. The final step is publishing a feature service to the ArcGIS Server. The current versions ArcPy supports only publishing of map services and not feature services. Hence this part is done manually through ArcMap.

The service accessed in web is as shown in Figure 16. It has the feature layers, attributes and symbology. It is consumed by the mobile client service through its REST URL.

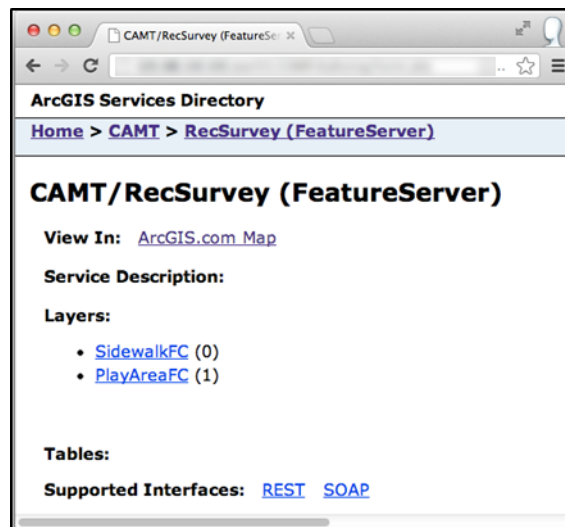


Figure 16. Published feature service

4.5 Implementation of Mobile Application

Implementation of the mobile application is not specific to the case study assessment as the app is built once and all the assessments fall within it. An iOS application, which can be run on iPhone and iPod Touch is built. Software used are Objective C using XCode IDE and ArcGIS API for iOS.

4.5.1 View Controllers in an iOS App

An iOS application is made up number of view controller classes. They are the controller objects in the Model-View-Controller (MVC) design. They also form the vital link between the app's data and its visual appearance ("View Controllers," n.d.). There are several in-built view controllers that are used in the application. A view controller may have multiple views and can reference other views. It can have gesture recognizers and other user interface objects. Some examples of view controllers used in the development are Table View Controllers, Navigation Controllers and Web View Controller. Various custom view controllers were also created for the app.

A storyboard in an iOS application holds preconfigured instances of view controllers and their associated objects. Storyboards make configuring the objects straightforward, so that the programming can focus on how the objects should behave. Instances of classes like UIView, UIButton etc. are then created on view controllers.

4.5.2 ArcGIS API for iOS

ArcGIS API for iOS helps us to build applications utilizing the mapping, geocoding, geoprocessing and other capabilities provided by ArcGIS server ("ArcGIS iOS SDK," n.d.). As ArcGIS server is the primary geographic information system that would be used by the assessments and its API is optimal for developing the mapping module of the app. The API includes maps, layers, graphics and tasks.

4.5.3 Implementation of Management Module

Figure 17a shows the user login screen. It is implemented by a custom View Controller (VC) inherited from UIViewController. If the user closes the app without

logging out, upon reopening, the app skips this screen. When a user logs in, an entry is made into the local file database through an instance of `NSUserDefaults` class and it persists the user login upon closing the app. The application's `UIApplicationDelegate` protocol monitors events for various application states including start up and termination. The methods of this protocol checks `NSUserDefaults` and maintains user session if he is already logged in.

Figure 17b shows the list of assessments available to the user. It is implemented as a custom VC inherited from `UITableViewController`. There are other assessments in addition to *RecSurvey*.



Figure 17. Mobile app – login & assessment

a) Login Screen b) List of Assessments



The user then has the choice to select the projects that he has access to. This is also a class that is inherited from UITableViewController. If the user knows the project ID for another project, he might add it. He can also remove projects that he no longer would participate in. These changes are updated to the database. This is shown in Figure 18a.

Figure 18. Mobile app - Project & Map editing mode
 a) List of projects in edit mode b) Empty Map in edit mode

4.5.4 Implementation of Mapping Module

On choosing the project, the map view loads which is an instance of AGSMapView of ArcGIS API. The location of project is zoomed. As shown in



Figure 19. Mobile app - Layers & Geometry
 a) Available feature layers to select b) User creating point

Figure 18b, an empty map is displayed, as there are no features to start with. When a new feature is created, a new custom view is created and shown in Figure 19a. It lists all the layers and types of features inside each layer that are part of this assessment. These are displayed from the REST end point. Symbology is imported from ArcGIS Server. Once the user selects which feature to collect, the view goes into editing mode. Touch event is changed. In order to add a point, the user just clicks on the map as shown in Figure 19b; to change a point, she just clicks somewhere else. In case of line or geometry, the app behaves accordingly.

After the geometry is created, the app is out of editing mode, and the default touch mode becomes active. Then the page for collecting non-spatial attributes loads. When an existing feature is clicked, a call out is displayed as shown in Figure 20a. Then the user can edit the feature both geometry and attributes. If some one else had created the feature, she can only view the attributes collected. The read only attributes are displayed using ArcGIS API's PopupViewController.



Figure 20. Mobile app - Call out & Survey

a) Map view showing features and call-out b) Dynamic decision tree questionnaire

4.5.5 Implementation of Web Module

When the user creates new features or edits the features she created earlier, she inserts non-spatial attributes. This is done by the questionnaire created as described in

Section 4.3.2.1 earlier. The questionnaire is presented in a custom class, which is an instance of web view controller. This is shown in Figure 20b. A web view controller embeds a web page into a view within the mobile app. The unique *Object ID* of the feature whose geometry was just created on the map, the assessment id and feature class id are passed as HTTP parameters to the web page because the correct XML needs to be loaded. As the name of the XML has the latter two values, the page creates the correct XML name, locates it in a fixed file location in the server and loads the questionnaire. Once all the questions are answered, and the form is validated, the attributes are updated into the appropriate fields in ArcGIS server for that *Object ID*. When editing an existing feature, the attributes are overwritten in the spatial database.

4.6 Evaluation and Rating of Assessments

The data is collected and stored over feature services. Once the data is collected, there might be a need to assess the quality or provide ratings to the features collected. For evaluating the quality, a separate assessment can be created, and fewer users can be granted access. A separate assessment can also be created for rating the features, for instance, the service at restaurants, or cleanliness of park. Both the types of assessments are published using map services, there by holding the features read-only.

While such an evaluation or rating assessment is not automated with authoring and GIS services creation, which is a scope for future, it is integrated into the mobile app created for data collection.

Configuration XML files are created manually or through any XML creation tool. The XML is parsed to include a fixed set of questions. For example, if a restaurant has to be rated with questions about service, quality, ambience, and overall rating from very bad to very good, an XML file is created to suit this unlike in data collection assessment where the questions were dynamic.

In the app, the rating assessment is as shown in Figure 17b. Projects are similar to data collection assessment. This map interface is also programmed using ArcGIS API for iOS and is simpler than data collection. The restaurant features already exist and are loaded from existing published map service REST end points. Therefore they are read-



Figure 21. Mobile app - Rating of an assessment
a) Listing of restaurants b) Survey questionnaire

only. On clicking a feature, a call out is displayed as shown in Figure 21a. On clicking the disclosure arrow, the static jQuery mobile page loads in a web view controller as shown in Figure 21b. Again, the object ID of the restaurant is passed as a HTTP parameter to this page so that the answers are entered for the correct restaurant in the database. Simple analysis like average rating can be done on this data.

CHAPTER 5. CONCLUSION AND FUTURE WORK

5.1 Conclusion

In this thesis, we present a framework that facilitates fast deployment of voluntary geographic information data collection assessments. The main steps involved in the process are to gather meta data related to the assessment, create and publish GIS services, consume the mapping services in a client mobile app, and present a dynamic decision tree questionnaire to collect non-spatial attributes.

A web-based assessment creation tool is developed and is used by administrators with domain knowledge. The admins enter the following: GIS metadata to create the feature classes, attribute details for each feature class and optional read-only map services. A hierarchy based XML model is designed and implemented to represent the order in which the attributes are to be collected from the user and it takes the relationship between attributes into account. By parsing the XML, a dynamic decision tree is created.

A geoprocessing service is created and published to the GIS server to automate creating and publishing the feature classes. The automation process includes the following: creation of feature classes in a spatial database engine, grouping the feature classes into a feature set, creating a map document and including the feature classes and validating the map document. The map document is then published as both map and feature service to the GIS server.

An iOS mobile application is designed and implemented for field data collection. An assessment created and published as mentioned above is accessible on the mobile application. Thus a single mobile application can be used to host multiple projects in

multiple locations. A VGI participant in the field uses the mobile app to collect data for any number of assessments and projects that he has access to. The feature layers for each assessment are displayed on the app and the user can choose one based on the feature she or he needs to map. Point, line and polygon geometries can be collected by this app. The user can edit the earlier features created by him or her and can view features created by others. Once feature geometry is collected, non-spatial attributes for that feature are collected through a custom-developed dynamic questionnaire embedded in a web form in the app.

5.1.1 Administrator Benefits

In this framework, assessment creation is simple and robust. It is also done rapidly within minutes when compared to traditional way of creating an assessment. Also as we demonstrated in Section 4.6, an evaluation assessment was implemented in addition to data collection. Thus the framework is highly extendable and scalable. A developer's work is also greatly simplified, as most components are reusable.

5.1.2 User Benefits

The framework exposes a single user interface for various assessments, ensuring consistency and ease of use by participants. Components displayed on the mobile app are in most cases retrieved from the server or from a web service; hence changes to the app can be made on the server side with minimum or no changes to the app itself. The human computer interaction issues like filling in redundant questions in the field and having an irrelevant static survey for all assessments are eliminated by the dynamic decision tree implementation. Thus all the research objectives put forth in the thesis are met.

5.1.3 Cost Benefits

In addition to the other benefits, the framework presented in the thesis also has significant cost benefits. An assessment is currently done using iPhones and the ArcGIS version of the tool by 24 volunteers who map a typically sized community of 2,000 people or the area 3/4 to 1 mile around a school over a course of 2 hours. If a community had the equipment and technology to do the work itself, the labor time to map the same area would be \$400 to \$500 less if a city staff member were to do the project. This is by calculating an intern salary of \$10 per hour. This is an extremely conservative price. By leveraging volunteers and their own smart phones, the city benefits greatly by not having to pay for the cellular data costs or the equipment. Also the cost benefits accrue when multiple assessments are carried out every year in multiple locations, as is currently carried out by Iowa State University's Extension.

5.2 Future work

We are currently migrating the existing assessments in ISU Extension into the framework proposed in the thesis. Ongoing work and future scope include the following:

- Tracking the users as mentioned in the framework can be implemented. The metadata is currently collected. GIS and client side development is to be done.
- *Evaluation* of services is demonstrated as an additional assessment that can be tied to the framework. Web authoring tool and geoprocessing automation for the same is to be implemented.
- Administrators can plan for many different types of assessments involving route planning, geocoding and so on.

- A more graphical assessment creation tool can be put in place; the tool can also allow the admins to edit the assessments.
- Automating of feature service publication is not automated due to software limitation. As new versions and service packs of the software are released, a fix can be put in place.
- Offline support for the mobile app can be developed in field conditions where there is no cellular reception.

BIBLIOGRAPHY

- ArcGIS Server. (n.d.). Retrieved April 11, 2012, from http://webhelp.esri.com/arcgisserver/9.2/dotnet/manager/concepts/whats_server.htm
- ArcGIS iOS SDK. (n.d.). Retrieved April 11, 2011, from <http://resources.arcgis.com/content/arcgis-iphone/api>
- ArcPy. (n.d.). Retrieved April 11, 2012, from <http://help.arcgis.com/en/arcgisdesktop/10.0/help/index.html#//000v000000v7000000>
- ArcSDE. (n.d.). Retrieved April 11, 2012, from <http://en.wikipedia.org/wiki/ArcSDE>
- Atlassian. (n.d.). Java Rosa. Retrieved from <https://bitbucket.org/javarosa>
- Butler, D. (2006). The web-wide world Home page, 439(February).
- Cocoa Architecture. (n.d.). Retrieved April 11, 2012, from <https://developer.apple.com/library/mac/#documentation/Cocoa/Conceptual/CocoaFundamentals/WhatIsCocoa/WhatIsCocoa.html>
- Cybertracker Conservation. (n.d.). Cyber Tracker.
- DataDyne. (n.d.). Epi Surveyor. Retrieved from <http://datadyne.org/>
- Dimagi Inc. (n.d.). CommCare. Retrieved from <http://www.commcarehq.org/home/>
- Elwood, S., Goodchild, M. F., & Sui, D. Z. (n.d.). Researching Volunteered Geographic Information : Researching Volunteered Geographic Information : Spatial Data , Geographic Research , and New Social Practice, (August 2012), 37–41.
- GIS. (n.d.). Retrieved from http://en.wikipedia.org/wiki/Geographic_information_system
- Geoprocessing. (n.d.). Retrieved from <http://en.wikipedia.org/wiki/Geoprocessing>
- Goodchild, M. (2007). Citizens as sensors: the world of volunteered geography. *GeoJournal*, 69(4), 211–221. doi:10.1007/810708-007-9111
- Graham, M. (2010). NEOGEOGRAPHY AND THE PALIMPSESTS OF PLACE: WEB 2.0 AND THE CONSTRUCTION OF A VIRTUAL EARTH. *Tijdschrift voor Economische en Sociale Geografie*, Vol.101(4), p.422(15).
- Harrison, W. (n.d.). Interactive Decision Tree. Retrieved from <https://github.com/hungrymedia/interactive-decision-tree>

- Hartung, C., Anokwa, Y., Brunette, W., Lerer, A., Tseng, C., & Borriello, G. (2010). *Open Data Kit : Tools to Build Information Services for Developing Regions*.
- Heipke, C. (2010). ISPRS Journal of Photogrammetry and Remote Sensing Crowdsourcing geospatial data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(6), 550–557. doi:10.1016/j.isprsjprs.2010.06.005
- Kamel Boulos, M. N., Resch, B., Crowley, D. N., Breslin, J. G., Sohn, G., Burtner, R., ... Chuang, K.-Y. S. (2011). Crowdsourcing, citizen sensing and sensor web technologies for public and environmental health surveillance and crisis management: trends, OGC standards and application examples. *International journal of health geographics*, 10, 67. doi:10.1186/1476-072X-10-67
- Lwin, K. K. (2011). Web-based GIS System for Real-time Field Data Collection Using Personal Mobile Phone. *Journal of Geographic Information System*, 03(04), 382–389. doi:10.4236/jgis.2011.34037
- Maué, P., & Schade, S. (2008). Quality Of Geographic Information Patchworks. *Proceedings of AGILE*, 1–8. Retrieved from http://plone.itc.nl/agile_old/conference/2008-girona/PDF/111_DOC.pdf
- Microsoft. (n.d.). Technology Model.
- Nielsen. (2012). Young Adults and Teens Lead Growth Among Smartphone Owners. Retrieved November 10, 2012, from http://blog.nielsen.com/nielsenwire/online_mobile/young-adults-and-teens-lead-growth-among-smartphone-owners/
- Nielsen, J. (2006). Participation Inequality : Encouraging More Users to Contribute Downsides of Participation Inequality.
- OpenStreetMap Stats. (n.d.). Open Street Map Stats. Retrieved from http://www.openstreetmap.org/stats/data_stats.html
- OpenStreetBugs. (n.d.). Open Street Bugs. Retrieved from <http://openstreetbugs.schokokeks.org/>
- OpenStreetMap. (n.d.). OpenStreetMap. Retrieved from <http://www.openstreetmap.org/>
- OpenStreetMap Wiki. (n.d.). OpenStreetMap Wiki. Retrieved from <http://wiki.openstreetmap.org/wiki/FAQ>
- Parikh, T., Javid, P., & Sasikumar, K. (2006). Mobile Phones and Paper Documents : Evaluating A New Approach for Capturing Microfinance Data in Rural India.

- Patnaik, S., Brunskill, E., & Thies, W. (2009). Evaluating the accuracy of data collection on mobile phones: A study of forms, SMS, and voice. *2009 International Conference on Information and Communication Technologies and Development (ICTD)*, 74–84. doi:10.1109/ICTD.2009.5426700
- Pendragon. (n.d.-a). Pendragon Forms. Retrieved from <http://pendragonsoftware.com/>
- Pendragon. (n.d.-b). Pendragon Forms case study.
- Pew Internet. (2011). Smartphone Adoption and Usage. Retrieved November 10, 2012, from <http://www.pewinternet.org/Reports/2011/Smartphones.aspx>
- Poser, K., & Dransch, D. (2010). Volunteered Geographic Information for Disaster Management with Application to Rapid Flood Damage Estimation, 89–98.
- RAD. (n.d.). Retrieved April 11, 2012, from http://en.wikipedia.org/wiki/Rapid_application_development
- Raento, M., Oulasvirta, a., & Eagle, N. (2009). Smartphones: An Emerging Tool for Social Scientists. *Sociological Methods & Research*, 37(3), 426–454. doi:10.1177/0049124108330005
- RapidSMS. (n.d.). Rapid SMS. Retrieved from <http://www.rapidsms.org/>
- Sabone, B. (2009). Assessing Alternative Technologies for Use of Volunteered Geographic Information in Authoritative Databases, (269). Retrieved from <http://gge.unb.ca/Pubs/TR269.pdf>
- Seeger, C. J. (2008). The role of facilitated geographic information in the landscape planning and site design process. *GeoJournal*, 72(3), 199–213. doi:10.1007/s
- Spatial Database. (n.d.). Retrieved from http://en.wikipedia.org/wiki/Spatial_database
- View Controllers. (n.d.). Retrieved April 11, 2012, from <http://developer.apple.com/library/ios/#featuredarticles/ViewControllerPGforiPhoneOS/Introduction/Introduction.html>
- Werts, J. D., Mikhailova, E. a, Post, C. J., & Sharp, J. L. (2012). An integrated WebGIS framework for volunteered geographic information and social media in soil and water conservation. *Environmental management*, 49(4), 816–32. doi:10.1007/s00267-012-9818-5
- Whitney, K., Batinov, G., Miller, L., & Ashenfelter, K. T. (2011). Exploring a Map Survey Task ' s Sensitivity to Cognitive Ability, (c), 63–68.

Yan, Y., Yu, J., Wu, J., & Ma, S. (2009). Design and Implementation of a Mobile GIS for Field Data Collection. *2009 WRI World Congress on Computer Science and Information Engineering, 1*, 235–241. doi:10.1109/CSIE.2009.538

Zhang, J., & Goodchild, M. F. (2002). *Uncertainty in geographical information* (2002nd ed.). London ; New York: Taylor & Francis.

Zhong, H., Li, P., Hu, Y., Lv, Z., Yin, J., Yu, B., & Wu, J. (2010). A solution for the data collection in the field survey based on Mobile and Wireless GIS. *2010 18th International Conference on Geoinformatics*, 1–5. doi:10.1109/GEOINFORMATICS.2010.5567747

jQuery UI. (n.d.). Retrieved from <http://jqueryui.com/>

kiwanja.net. (n.d.). Frontline SMS. Retrieved from <http://www.frontlinesms.com/>