

Development for Transportation Asset Management Inventory & Management Tools

FINAL REPORT
September 2014

Submitted by:

Devin Squire
Utah State University
Logan UT 84322

Kevin Heaslip
Associate Professor
Utah State University
Logan UT 84332

External Project Manager
Abdul Wakil
Utah Department of Transportation

In cooperation with

Rutgers, The State University of New Jersey

And

State of Utah

Department of Transportation

And

U.S. Department of Transportation

Federal Highway Administration

Disclaimer Statement

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the Department of Transportation, University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

1. Report No. CAIT-UTC-002		2. Government Accession No.		3. Recipient's Catalog No.	
4. Title and Subtitle Development for Transportation Asset Management Inventory & Management Tools			5. Report Date September 2014		
			6. Performing Organization Code CAIT/Utah State University		
7. Author(s) Devin Squire, Kevin Heaslip			8. Performing Organization Report No. CAIT-UTC-002		
9. Performing Organization Name and Address Utah State University Logan UT 84322			10. Work Unit No.		
			11. Contract or Grant No. DTRT12-G-UTC16		
12. Sponsoring Agency Name and Address Center for Advanced Infrastructure and Transportation Rutgers, The State University of New Jersey 100 Brett Road Piscataway, NJ 08854			13. Type of Report and Period Covered Final Report 4/1/12-10/30/13		
			14. Sponsoring Agency Code		
15. Supplementary Notes U.S. Department of Transportation/Research and Innovative Technology Administration 1200 New Jersey Avenue, SE Washington, DC 20590-0001					
16. Abstract <p>Mobile technology is becoming more and more pervasive within the consumer industry. Devices such as smartphones and tablets are able to relay information effectively and affordably either with or without an accompanying cellular connectivity plan. The effect of this technology is slowly making its way through the corporate world. Recognizing that this technology has the potential to affect workflow practices within a Department of Transportation, this report first presents two mobile applications to address specific areas of concern identified within the Utah Department of Transportation. The first application creates a living directory that provides specific information regarding employees, equipment, and location within maintenance shed locations throughout the state of Utah. The second application provides a seamless method of information transfer as it relates to traffic signs from the field directly to central servers in an effort to reduce data loss and corruption.</p> <p>In order to facilitate a determination of effectiveness, the second aspect of this report develops a framework through which the developed applications can be deemed as either effective or ineffective given indicators of fit, viability, usability, and productivity. These areas are assessed using a defined survey that presents users of the mobile applications with a series of questions relevant to each of the four indicators. The surveys are then analyzed through both descriptive and inferential statistical methods in order to provide a conclusion of effectiveness. Because they seek to develop a basic framework, the concepts and practices presented in this report can be used as a guide for the assessment of other mobile applications as they become available. As such, the application of this framework will allow agencies to confidently incorporate mobile technology into everyday work practices.</p>					
17. Key Words Asset Management, Inventory, Mobile Technology, Information Transfer,			18. Distribution Statement		
19. Security Classification (of this report) Unclassified		20. Security Classification (of this page) Unclassified		21. No. of Pages 168	22. Price

ABSTRACT

Mobile technology is becoming more and more pervasive within the consumer industry. Devices such as Smartphones and Tablets are able to relay information effectively and affordably either with or without an accompanying cellular connectivity plan. The effect of this technology is slowly making its way through the corporate world. Recognizing that this technology has the potential to affect workflow practices within a Department of Transportation, this report first presents two mobile applications to address specific areas of concern identified within the Utah Department of Transportation maintenance division. The first application creates a living directory that provides specific information regarding employees, equipment, and location within maintenance shed locations throughout the state of Utah. The second application provides a seamless method of information transfer as it relates to traffic signs from the field directly to central servers in an effort to reduce data loss and corruption.

In order to facilitate a determination of effectiveness, the second aspect of this report develops a framework through which the developed applications can be deemed as either effective or ineffective given indicators of fit, viability, usability, and productivity. These areas are assessed using a defined survey that presents users of the mobile applications with a series of questions relevant to each of the four indicators. The surveys are then analyzed through both descriptive and inferential statistical methods in order to provide a conclusion of effectiveness. Because they seek to develop a basic framework, the concepts and practices presented in this report can be used as a guide for the assessment of other mobile applications as they become available. As such, the application of this framework will allow agencies to confidently incorporate mobile technology into everyday work practices.

CONTENTS

ABSTRACT.....	i
LIST OF TABLES	vi
LIST OF FIGURES	viii
CHAPTER 1	1
1.1 Research Question	1
1.2 Research Problem	1
1.2.1 Maintenance Shed Information.....	1
1.2.2 Traffic Sign Data.....	4
1.3 Mobile Development	5
1.3.1 Maintenance Sheds	6
1.3.1.1 Automatic Filtration.....	6
1.3.1.2 Manual Filtration	6
1.3.1.3 Location Information	6
1.3.2 Traffic Signs.....	7
1.3.2.1 New Data Entry.....	7
1.3.2.2 Record Maintenance Activities.....	7
1.4 General Approach to Analysis Framework.....	8
1.5 Research Outline.....	8
CHAPTER 2	9
2.1 Purpose.....	9
2.2 Approach to Traffic Sign Management	9
2.2.1 Goals of Asset Management	9
2.2.2 Benefits of Asset Management	10
2.2.3 Asset Management Methodology	11
2.2.4 Creating an Inventory Database.....	12
2.2.4.1 Manual Data Collection	12
2.2.4.2 Mobile Data Collection.....	13
2.2.4.3 Data Elements	14
2.2.5 Maintaining the Inventory.....	15

2.3 Development and Implementation of Mobile Technology	17
2.3.1 Advantages to the Implementation of Mobile Technology	18
2.3.2 Fundamental Objectives in Developing a Mobile Solution.....	18
2.3.3 Success Factors	20
2.4 Process Evaluation	23
2.4.1 Usability	23
2.4.2 Productivity.....	24
2.4.3 MoBiS-Q.....	24
2.4.4 Likert Scales.....	28
2.4.5 Survey Creation	29
2.4.5.1 Optimal Survey Length.....	29
2.4.5.2 Survey Administration Method.....	29
2.5 Conclusion	29
CHAPTER 3	30
3.1 Introduction.....	30
3.2 Maintenance Shed Application.....	30
3.2.1 ShedVIEW Programming Efforts	32
3.2.1.1 Manual Search	34
3.2.1.2 Automatic Search.....	37
3.2.1.3 Location Specific Information	38
3.2.2 Database Programming Effort	39
3.3 Sign Management App	40
3.3.1 ODK Collect	40
3.3.2 XLSForm	41
3.3.3 ODK Aggregate	44
3.4 Conclusion	47
CHAPTER 4	48
4.1 Introduction.....	48
4.2 Sample Groups.....	48
4.2.1 Managers.....	49

4.2.2 Engineers.....	49
4.2.3 Field Personnel.....	49
4.3 Survey Design.....	49
4.3.1 Survey Development.....	50
4.3.1.1 MoBiS-Q for ShedVIEW.....	50
4.3.1.2 MoBiS-Q for the Data Collection App.....	51
4.3.1.3 Questions to Determine Viability and Fit.....	52
4.3.2 Final Survey.....	54
4.3.3 Determining Viability.....	57
4.3.4 Survey Administration Timetable.....	60
4.4 Survey Result Analysis.....	60
4.4.1 Data Preparation.....	60
4.4.1.1 Cleaning Data Set.....	61
4.4.1.2 Correcting Data.....	62
4.4.1.3 Defining Data Categories.....	63
4.4.2 Data Analysis Using Likert Scales.....	63
4.4.3 Analysis Framework.....	64
4.4.3.1 Frequency Distribution.....	65
4.4.3.2 Additional Descriptive Statistics.....	67
4.4.3.3 Viability Survey Analysis.....	68
4.4.3.4 Interpreting Results.....	70
4.4.4 Statistical Inference Framework.....	71
4.4.4.1 Goodness of Fit Test.....	71
4.4.5 Sample Size.....	72
4.5 Feedback Loop.....	73
4.6 Conclusion.....	74
CHAPTER 5.....	75
5.1 Research Question Readdressed.....	75
5.2 Potential Conclusions from Analysis Framework.....	76
5.3 Future Efforts.....	76

5.3.1 ShedVIEW Efforts	77
5.3.2 Data Collection Application Efforts	78
5.3.3 Non-Specific Efforts	78
REFERENCES	80
APPENDICIES	83
Appendix A : ShedVIEW User Manual.....	84
CHAPTER 6	85
Appendix B : Sign Data Collection Application User Manual.....	99
Appendix C : ShedVIEW Development Documentation	108
Appendix D : Sign Data Collection Application Development Documentation	133
D.1 Collect	134
D.1.1 Programming ODK.....	134
D.1.2 Licensing Restrictions.....	137
D.1.3 Reference Web Sites	138
D.2 Aggregate.....	139
D.2.1 Deployment.....	139
D.2.2 Connecting Collect Devices to Aggregate Instance.....	141
D.2.2.1 Setting up Aggregate for Submissions.....	141
D.2.2.2 Pointing ODK Collect to Aggregate Deployment	142
D.2.3 Multiple Users and Data Retrieval from Aggregate	143
D.2.4 Uploading Forms.....	144
D.2.5 Exporting Data	144
D.2.6 Future Work	145
D.3 Form Builder	146
D.3.1 Form Template.....	146
D.3.2 Syntax.....	146
D.3.3 Converting Excel Form to XML.....	147
D.3.4 General Notes.....	147
Appendix E : UDOT Sign Management OMS Fields	149
E.1 Element 410 – Sign Support.....	150
E.2 Element 410 – Sign Face.....	154

LIST OF TABLES

Table		Page
2.1	Core Data Elements.....	14
2.2	Critical Data Elements.....	15
2.3	Desirable Data Elements	15
2.4	MoBiS-Q Themes and Questions with IT Survey Reference	27
3.1	ODK Question Types	42
3.2	Metadata	43
4.1	MoBiS-Q for ShedVIEW	51
4.2	MoBiS-Q for the Data Collection App.....	52
4.3	Statements to Determine Viability and Fit	53
4.4	Response Conversion Values (Likert).....	61
4.5	Response Conversion Values (Demographic).....	62
4.6	Questions Requiring Correction	63
4.7	Response Correction Values	63
4.8	Sample Data for ShedVIEW Survey	65
4.9	Sample Data Statistics	68
4.10	Responses to the ShedVIEW Viability Survey	69
4.11	Viability Data Statistics.....	69
4.12	Chi-Square Calculated Values.....	72
4.13	Chi-Square Expected Values.....	72

D.1	Suggested String Changes.....	136
D.2	Applicable Web Sites.....	139
E.1	Sign Support Database Elements.....	150

LIST OF FIGURES

Figure		Page
1.1	Out of the Way UDOT Maintenance Shed	3
1.2	Detail of State Roads Crossing Region Boundaries	4
2.1	DOT Information and Money Flow Diagram	11
2.2	Relationship of Components in Asset Management System.....	12
2.3	Integrated Traffic Sign Management System.....	17
2.4	Procedures of Value-Focused Thinking	19
2.5	Means-Ends Objective Network for a Mobile Application.....	20
2.6	Fit-Viability Framework.....	21
3.1	ShedVIEW Mind-Map	32
3.2	ShedVIEW Home Screen	36
3.3	ShedVIEW Search by Region	37
4.1	Effectiveness Indicators	48
4.2	ShedVIEW Final Survey	55
4.3	Data Collection App Final Survey	56
4.4	Viability Survey	59
4.5	Frequency Distribution for Sample Data.....	67
4.6	Box and Whisker Plot for Sample Data	68
4.7	Frequency Distribution for Viability Data	70
4.8	Box and Whisker Plot for Sample Data	70
4.9	Feedback Loop	74

5.1	Research Approach.....	76
D.1	Cloning an Application	135
D.2	Monitor Tomcat Dialog.....	141
D.3	Aggregate Retrieval Setting	144

CHAPTER 1 INTRODUCTION

In recent years, transportation agencies have been hit with budget constraints and increased accountability. Agencies have been left to find ways to cut and save while still fulfilling their responsibilities to the public while facing increased public scrutiny. Recent advances in the field of mobile technology hold promise to help stricken agencies, providing low cost tool options for improving efficiency while using equipment already in the hands of employees.

1.1 Research Question

The purpose of this research is to develop a framework that can be used in determining the effect of incorporating mobile technology into a state Department of Transportation (DOT). More specifically, this research investigates potential mobile solutions to two distinct challenges relating to the transfer of information maintained on centralized databases within a DOT. It then presents the development of potential solutions, and finally develops a framework for analyzing the implementation of the technology's effects within an agency.

1.2 Research Problem

This research specifically seeks to address two challenges that the Utah Department of Transportation (UDOT) currently faces. The first challenge addresses a living database of information relating to equipment and asset holdings that are controlled by UDOT. The complexity of this information provides an opportunity to utilize mobile technology. The second challenge addresses the issue of traffic sign asset management where a field technician is required to record information relating to any maintenance performed on that specific asset and propagate that information to UDOT's Operations Management System (OMS).

1.2.1 Maintenance Shed Information

UDOT personnel have identified a need for the distribution of up to date information throughout the maintenance system that is both useful and timely to the organization as a whole. This information can be divided into three categories, namely information relating to the central office, and maintenance sheds, as well as regional information used to help a user identify

specific location-based values. While most of this information is available within UDOT servers, accessing and distributing the information as well as keeping the information up to date has proven to be challenging.

UDOT controls 96 maintenance stations and substations in 4 different regions of Utah. These maintenance stations are scattered throughout the 5,949 miles of state highways within Utah (1). Each maintenance station controls an inventory of equipment assets meant to facilitate work done to the highway systems. This equipment includes 498 trucks equipped with snowplows, as well as 1100 other pieces of specialized equipment.

In order to maintain their asset holdings, UDOT employs 560 full time employees and 65 seasonal employees (1).” The sheer amount of both personnel and equipment asset information that exists on UDOT servers presents a challenge to up to date distribution. When personnel changes and movements as well as the acquisition or dispersal of equipment is taken into consideration, this information becomes extremely complex due to its dynamic nature.

While constantly changing data provides UDOT employees with a challenge, it is not the only issue that the implementation of technology can address. Many maintenance sheds have been historically hard to find due to their out of the way locations, as shown in Figure 1.1. Time is wasted by employees because of this challenge in trying to locate sheds. Additionally, not all maintenance-shed locations have all types of assets (i.e. fuel). The need has been expressed to identify specific shed locations that would be of the most benefit to the interested party. Maintenance personnel in the field would benefit from a way to identify shed locations within their general area should the need arise for refueling, obtaining materials, acquiring specific equipment, etc.



FIGURE 1.1 Out of the Way UDOT Maintenance Shed (2)

Another area where the implementation of mobile technology could help to increase productivity is in finding location specific information. Maintenance workers in the field record work performed specific to a given location. The addition of accurate location data regarding work activity including the county, UDOT region, milepost, and the state route on which maintenance work is performed would be of use to a maintenance worker in the field. While this data is generally intuitive, the need for definite location is highlighted when state routes cross UDOT region borders, as shown in Figure 1.2. This figure details three main roadways which cross UDOT region boundaries in approximately the same location. A UDOT employee would benefit from knowing that their current location was either slightly within, or slightly without their region's borders.



FIGURE 1.2 Detail of State Roads Crossing Region Boundaries (3)

1.2.2 Traffic Sign Data

Within the 5,949 miles of state highways within Utah, UDOT specifically accounts for approximately 95,000 post-mounted traffic signs (1). Due to the sheer quantity of signs, this population is inherently difficult to manage. Additionally, according to the National Cooperative Highway Research Program (NCHRP), “recent changes to the MUTCD with respect to traffic sign retroreflectivity have helped to make it necessary to better track the deterioration status of the entire traffic sign population (Re and Carlson 2012).” With increased public and federal scrutiny, it is necessary to build a better system of managing and maintaining the traffic sign population.

Previous efforts to gather information on the total sign population resulted in a total of 2500 signs and supports over the entire state of Utah (4). Data is gathered by maintenance personnel and then stored in UDOT’s Operations Management System (OMS) which manages data at a central location. For the 2500 signs and supports, data collected and stored in UDOT’s OMS included the following:

- MUTCD Codes
- Sign Legend

- Sign Dimensions
- Sign Face Material
- Sign Backing Material
- Sign Illumination
- Sign Support ID
- Sign ID

Also included in the OMS database are attributes regarding the type of support, support material and dimensions, and how the support is fastened to its base. Location information, such as the route number and milepost, and sign face orientation is also stored in the OMS (4). While there is a provision for all of this information to be stored in the OMS, historically only about 2.5% or 2500 out of an estimated 95,000 signs were recorded. The information within the database also varies in accuracy and thoroughness. In the data recorded for the 2500 signs, predominately lacking from the OMS is a record of sign installation, a factor that is used in many different ways save both time and budget commitments by a DOT.

One of the issues contributing to this lack of information has to do with field work documentation. Information that is recorded is either stored on a mobile laptop computer, on paper, or in a worker's memory. Information regarding maintenance work performed on signs also faces this same problem. If a sign is replaced due to damage, the changes may or may not make their way into the UDOT OMS. This results in an inaccurate and effectively unusable database. Mobile technology has a high potential to aid the transmission of data both to and from devices used directly in the field.

1.3 Mobile Development

The attempted solution to both of the problem areas mentioned in the previous paragraphs will be addressed with the development of two separate mobile applications. These applications will be built to work on any type of mobile device running Google's Android operating system with future possible extension to Apple devices. The applications will be developed in an effort to provide a user-friendly experience that can be efficiently used on devices that many UDOT personnel already carry in the field.

1.3.1 Maintenance Sheds

The first mobile application to be developed as part of this research will address problems currently faced by UDOT employees in keeping track of data relating to maintenance sheds. In general terms, the goal of this application is to enable a UDOT employee to be able to search through an up to date directory of information. This directory will be able to be accessed either automatically based on a personnel's location or manually through filtering content. It will also allow UDOT personnel to view specific information based on user location.

1.3.1.1 Automatic Filtration

Given this application, a UDOT employee would be able to use a mobile device in order to locate a maintenance shed closest to his or her current location. Upon selecting the "closest shed" option, a user would be presented with a map allowing selection of a desired maintenance shed. Alternatively, this information can be presented in a list type interface depending on user preference. A list interface would display maintenance shed locations and their proximity to the current location of the mobile device. Both the map and the list interface will display information about certain assets that are only found at certain locations, such as fuel. Upon selection of a maintenance shed from either interface, location specific information such as contact information, equipment available, and employee contact information will be displayed.

1.3.1.2 Manual Filtration

To aid personnel who need to search for areas not currently near their location, an option to manually search through maintenance sheds is given. This benefits employees needing to get in touch with a specific shed, or quickly access up to date employment or equipment information. Manual functionality allows an employee to quickly drill down by region to a specific maintenance shed's information. Information presented upon the completion of this selection process will be identical to the information presented in the automatic search option.

1.3.1.3 Location Information

The final portion of this application will provide information specific to a mobile devices location. This option will be extremely simple to use, requiring only selection of the option and prompted activation of the devices' global positioning system (GPS). Information relating to the

UDOT region and State County that are specific to the inquiry location will be returned. Additional information that can possibly be returned is still being explored.

1.3.2 Traffic Signs

In an attempt to find a better way of managing UDOT traffic sign data, a second mobile application will be developed. This application will allow personnel to record field data directly to a UDOT controlled database. Built into this application process will be checks and balances to ensure that data is entered correctly into the software, and uploaded to the server correctly. Aside from these checks and balances, the application will require functions for new data entry, and data modification.

1.3.2.1 New Data Entry

One of the primary functions of this mobile application will be to create new data entries within UDOT databases in the instance of new sign installation. Each time a new asset is installed, information such as location, size, retroreflectivity, etc will be input into the graphical user interface (GUI) of the application by maintenance personnel who perform the installation. This information will be entered into the application at the site of installation in an effort to minimize data discrepancy. A unique id will be assigned to the asset by the application and the data instance, upon mobile internet connectivity, will be uploaded to the main UDOT database.

1.3.2.2 Record Maintenance Activities

In addition to creating a new entry, a provision needs to be in place for data modification in the case of traffic sign repair and/or data collection activities. In this instance, a maintenance worker would be required to record information in the field after performing duties relating to the specific sign. Upon completion of maintenance activities, a new data entry would be created as described above. This instance however would prompt for an asset identification number, something that is unique to the sign that is being repaired. A maintenance worker would input this number and update the remainder of the fields as with the new data entry. Upon completion of entering the data entry, a save option will be presented. Upon confirmation, the data instance will be uploaded to UDOT servers for verification and recording. Background services would then be run to either overwrite or add the updated value in to the database. The checks and

balances mentioned above become imperative in the case of data modification to ensure an accurate database is maintained.

1.4 General Approach to Analysis Framework

A framework for analysis will be developed built upon survey-derived data. Once the applications are implemented within UDOT, users will be surveyed to determine the application acceptance and effectiveness. Measures of productivity will also be included in the analysis in an effort to provide a further description of the applications effect.

While the incorporation of mobile applications can generally be hypothesized as a solution in order to improve efficiency, the opposite can also be true. Mobile technology could potentially prove to be a distraction and essentially counter any productivity gained. Additionally, in the case of a poorly developed application, the technology might be too much of a hassle or may take too long to use compared to other possible methods of performing the same task.

1.5 Research Outline

The intent of this research is to report on efforts made in an attempt to address the effects of mobile technology within a DOT. Chapter two will provide a literature review of research that has already been performed on the development and implementation of mobile applications for productivity enhancement as well as on other related themes. Research on asset management and process evaluation will also be incorporated into this review. Chapter three will describe the mobile development process. Chapter four will formulate an analysis framework that can be used to determine if the implementation technology developed in chapter three was successful. Finally, Chapter five will present a conclusion and discuss the possibility of future research opportunities.

CHAPTER 2 LITERATURE REVIEW

2.1 Purpose

The intent of this literature review is to provide a base of research already performed on the topics of both traffic sign asset management and mobile technology. The review will be broken down into three categories in an effort to best reflect upon the problem areas that were discussed previously. The first section of review will provide an examination of research relating approaches in traffic sign management. Second, research relating to the development and implementation of mobile technology will be presented. In the third section, research will be presented relating to process evaluation as it relates to mobile devices.

2.2 Approach to Traffic Sign Management

An asset management system's goal is to "provide information for deciding whether to maintain, rehabilitate, or replace (5)." Traditional asset management allows agencies to make these decisions through a process of creating and maintaining an inventory, monitoring asset condition, and taking measures to extend an assets useable life. To be effective, the entire process must remain at an overall cost that is effectively less than that of asset replacement.

2.2.1 Goals of Asset Management

Because traffic signs are a first and foremost an asset to a DOT, a discussion of the goals that an asset management program aims to reach is merited. Asset management "combines engineering and mathematical analyses with sound business practice and economic theory (6)." Its goals, according to AASHTO's Transportation Asset Management Guide (7) are to:

- "Build, preserve, and operate facilities more cost-effectively with improved asset performance;
- Deliver to an agency's customers the best value for the public tax dollar spent; and
- Enhance the credibility and accountability of the transportation agency to its governing executive and legislative bodies."

2.2.2 Benefits of Asset Management

The benefits of asset management can be looked at in many ways depending on the agency being analyzed. Different methods in management philosophy, resources, priorities, and the agency's transportation system all account for this benefit diversity. Generally, asset management systems lower long term preservation costs, improve asset performance and service, improve cost-effectiveness, turn focus to performance and outcomes, and improve accountability in decisions and expenditures (Cambridge Systems, Inc et al. 2002).

These benefits can all be summed up in a financial umbrella. This can be seen at both the state and the local level in that information is a precursor to obtaining money. Information is gathered and then presented to financial decision makers. Funding is then allocated based upon the information presented. This process is best represented by the Federal Highway Administration (FHWA) in Figure 2.1 below. Asset management enables an agency to better understand and communicate and act on their financial responsibilities.

In the specific case of traffic signs, there are many specific benefits to asset management. Re and Carlson provide specific examples where benefits exist:

One of the advantages to sign inventory is the ability to quickly access sign data. One [study] participant reported that its system was frequently used when there was an inquiry from a concerned resident. Staff could promptly access specific sign information and promptly address the issue. Another example involved missing signs; a technician in the field would report any missing signs to the main office. The office would directly identify the missing sign and process the work order. In this way, the field technician would not be required to spend much time investigating and the replacement process could be accelerated.

The last example deals with knockdown signs. Before creating a sign inventory, one agency made two separate trips to handle such incidents. The first trip would ascertain the sign type, dimensions, and necessary hardware and the second trip was for the actual replacement or repair of the sign. Having a sign inventory eliminates the first trip, because a technician can determine all of the sign information from an office computer. Quickly accessing sign information is valuable and particularly beneficial when time is an issue (8).

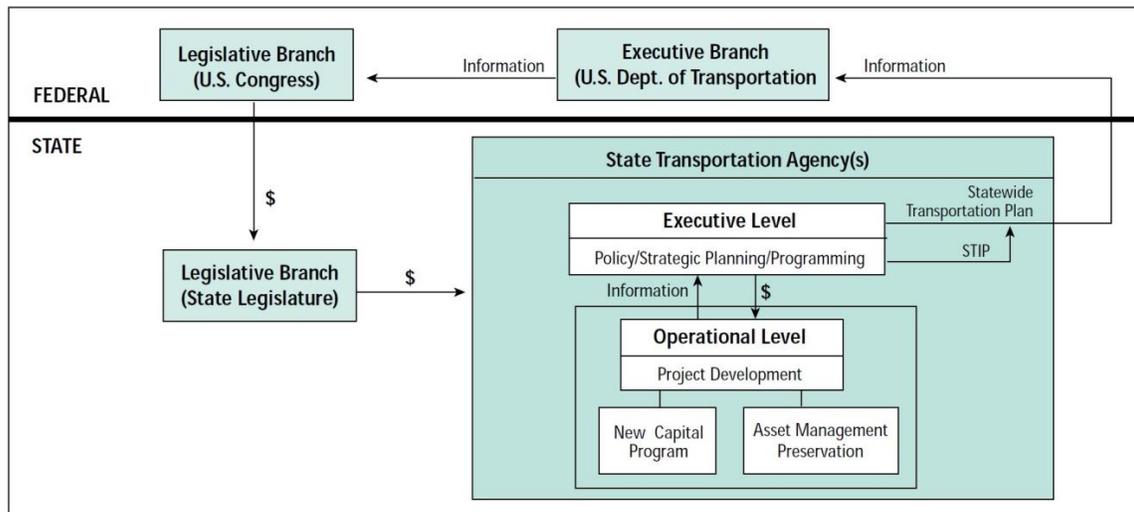


FIGURE 2.1 DOT Information and Money Flow Diagram (9)Asset Management

Methodology

According to the FHWA Asset Management Primer (9), elements of an asset management system include:

- “Strategic goals;
- Inventory of assets (physical and human resources);
- Valuation of assets;
- Quantitative condition and performance measures;
- Measures of how well strategic goals are being met;
- Usage information;
- Performance-prediction capabilities;
- Relational databases to integrate individual management systems;
- Consideration of qualitative issues;
- Links to the budget process;
- Engineering and economic analysis tools;
- Useful outputs, effectively presented; and
- Continuous feedback procedures.”

An asset management system may contain some or all of the elements above. To improve clarity, these components are summarized into a flowchart produced by the FHWA.

This chart, displayed in Figure 2.2, details key elements of an asset management system and how they relate to each other. As seen from the flowchart, the process of developing an effective asset management program may require an iterative process to best produce meaningful information.

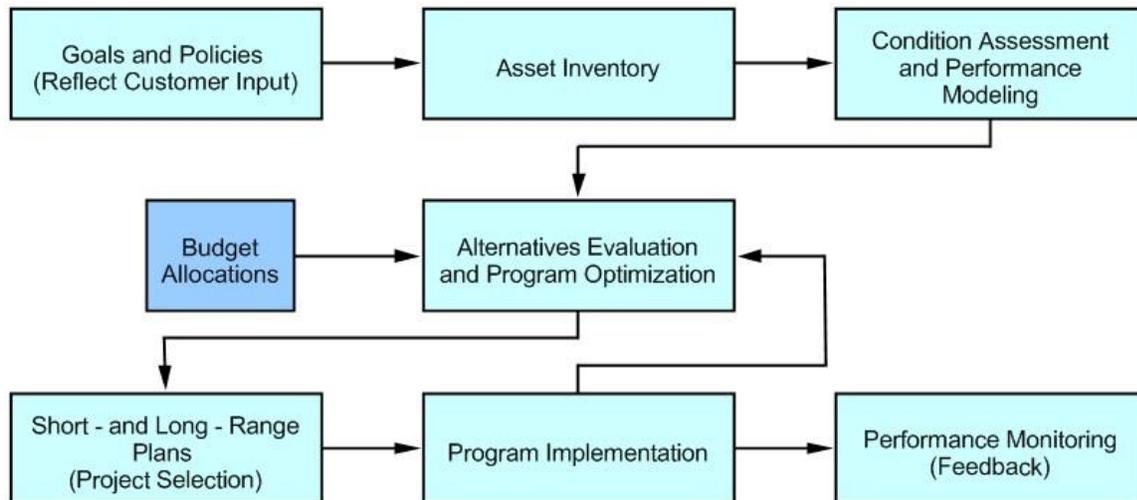


FIGURE 2.2 Relationship of Components in Asset Management System (6)Creating an Inventory Database

Creating an asset inventory is likely the hardest aspect of an asset management system. Data collection, which can be done either manually or by the use of a mobile collection unit, is an indispensable component to the asset management program.

2.2.4.1 Manual Data Collection

Traditionally, data collection has been done manually by personnel walking along the roadway taking measurements. This required simple equipment, little training, and provided good data. It is a slow method, however, with crews being able to cover a limited number of miles per day depending on the amount of information required (10). There is also an issue of safety to be considered with a manual collection method. This method often will require “data collectors to be in or near the roadway, or cross the road under extreme traffic conditions. Obviously, movement of persons in or near the roadway is not safe (11).”

2.2.4.2 Mobile Data Collection

Since manual collection tends to take a large amount of time, agencies are continuing to look to mobile data collection as alternative. Mobile collection started years ago with companies simply fitting a vehicle with cameras to record information to be analyzed at a later time in a lab setting. With recent advances in technology, today's mobile asset data collection vehicles are able to collect immense amounts of data at speeds equal to surrounding traffic (10). The capabilities of mobile data collection vehicles include:

- “High resolution still and video cameras;
- Laser scanners;
- Inertial navigation systems;
- Global positioning systems;
- Distance measurement instrument; and
- Retroreflectometers (11).”

These instruments allow analysis of a variety of assets including barriers, shoulders, signs, rumble strips, etc.

Mobile data collection continues to evolve in order to provide the greatest amount of asset information possible. While this collection method appears to be superior to manual collection, the accuracy of mobile collection needs to be addressed to make a clear decision between the two. This comparison was the topic of research performed by North Carolina State University in an article entitled “Comparison of mobile and manual data collection for roadway components.” The purpose of this research was to compare the collection quality of various mobile data collection companies to perfect data collected manually. After much analysis the authors concluded that “mobile data compared reasonably well to manual data for most... variables (11).” Certain assets, however, had a much lower collection quality than others. While in most cases mobile collection is superior to manual collection, in specific instances the opposite may prove to be true. In the consideration of implementing a mobile data collection process, the track record of each type of asset data to be collected must be analyzed in regards to mobile collection compared to manual collection (10).

2.2.4.3 Data Elements

It is important to know what data is and is not important when implementing a data collection method or maintenance system. The FHWA provided guidance regarding this selection, separating desirable information into three different categories: Core, Critical, and Desirable (12).

Core elements include information that is indispensable to the asset inventory. This information can be used to combat tort liability as well as provide data that can be used for limited asset management and budgeting (12). These core elements are presented in Table 2.1.

Critical elements provide a significant value to a sign inventory. While these elements will not necessarily provide information to combat tort claims, the addition of detail that comes from critical information can provide an agency with information to better target signs for replacement and perform cost analysis (12). Table 2.2 provides a listing of the critical data elements.

The final category of data is grouped together as desirable data elements. These elements can potentially add value to the inventory depending on the needs of an organization. These elements provide information above and beyond the information provided by both the core and critical data elements (12). A list of desirable elements can be found in Table 2.3.

TABLE 2.1 Core Data Elements (12)

Data Element	Description
Location	Includes several variables (such as route name, distance, etc.) depending on the location reference system that is selected.
Position	Location of the sign relative to the road (e.g. left, right, overhead, median).
Sign Code	Usually based on the MUTCD designations, may be supplemented or modified based on State or local sign designations.
Sign Condition	An assessment of the quality of the sign based on daytime and nighttime visual inspections.
Maintenance Activity	Maintenance activity associated with a particular sign.
Inspection/Maintenance Date	Date when the sign was inspected or maintained.

TABLE 2.2 Critical Data Elements (12)

Data Element	Description
Installation Date	Date when the sign face was installed.
Sign Size*	The width and height of the sign.
Sheeting Type	Grade of retroreflective sheeting.
Backing Type	Type of sign blank material.
Post/Support Type*	Type of sign support used, may include breakaway characteristics.
Post/Support Condition	An assessment of the quality of the sign support.
Sign Orientation	Cardinal direction the sign is facing.
Traffic Speed¹	Speed limit on the roadway where the sign is located.

TABLE 2.3 Desirable Data Elements (12)

Data Element	Description
Offset	Distance from the edge of pavement.
Height	Height of sign above the level of the road at the edge of the pavement.
Retroreflectivity	Objective measure of the nighttime quality of the sign.
Inspector	Name or initials of the individual who inspected or maintained the sign.
Sign identification Number	A unique number identifying the sign.
Images	Visual images of the sign, either digitally captured or linked to a videodisc-based photo-log.
Comments	Supplementary notes about the sign installation.
Other Reference Numbers	Could include maintenance district, plan or contract numbers, etc.

2.2.5 Maintaining the Inventory

There is a significant investment required to obtain an asset inventory. This investment can easily be put to waste if the proper framework is not in place to keep it current. With the use of technology, this effort can be greatly simplified and only needs to be utilized by a DOT in

* Recommended for offsite work order processing if similar sign is to be erected

¹ Changes have been made to FHWA requirements and specifications since these element suggestions were published (1998). As a result, traffic speed is no longer applicable for inclusion as a data element.

order to be effective. This technology is not new; each DOT has some method for managing their sign assets. Most DOT's designate this as a Sign Management System (SMS), which is diagramed in Figure 2.3. Inventory maintenance should fit into the DOT's existing SMS in order to be most effective (12).

The SMS should incorporate provisions for new and updated data. "A critical aspect of [inventory maintenance] occurs in the scheduled monitoring of the transportation system to gather data on how the transportation network infrastructure is performing, compare performance results to intended targets or policy objectives, and provide feedback to individual stages of resource allocation and utilization to identify needed adjustments in policy, procedures, and criteria for future management cycles (7)." Additionally, in order to ensure usability, as well as compliance with Federal standards, some provision for condition monitoring needs to be built into an effective SMS. This monitoring should account for:

- "Condition of sign face – indications of major cracking, delamination or peeling or blistering of the retroreflective sheeting materials, missing message, etc.
- Discoloration, streaking, or fading of the sign face.
- Visibility of the sign – roadside vegetation or a new structure may be blocking the motorist's view of the sign.
- Dirt or other substance of the sign.
- Vandalism.
- Orientation and structural stability of the sign support system.
- Usefulness or appropriateness – some signs may no longer be needed and should be removed.
- Poor retroreflectivity level (12)."

The most important part of maintaining an inventory is developing a procedure that will ensure compliance throughout an organization. "The inventory administrator must identify all installation and maintenance activities (including those done under contract by private forces) that affect the data elements...and develop a process for ensuring that these activities are reflected in the inventory (12)." This will help guide each contributor to the SMS in a way that will keep the information accurate and up to date, thus creating a perpetually useful management system.

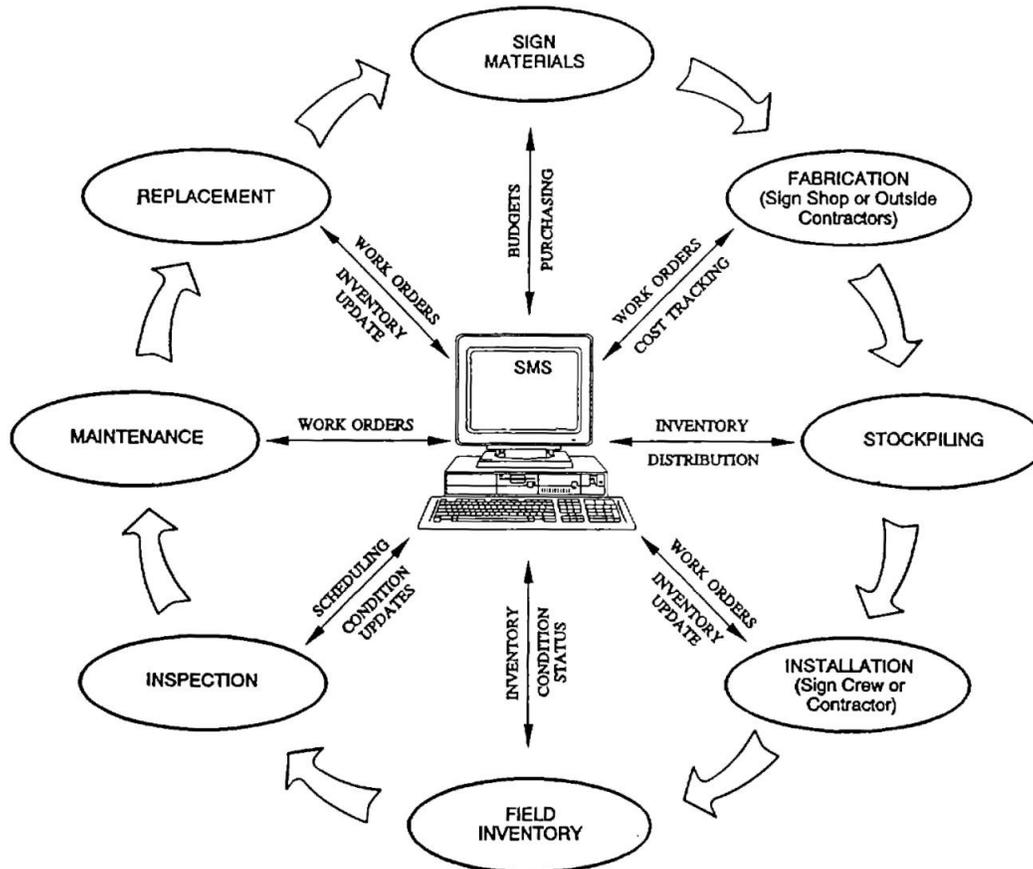


FIGURE 2.3 Integrated Traffic Sign Management System (12)Development and Implementation of Mobile Technology

Advances in the mobile technology market have helped to push such devices directly into the main stream of consumerism. The public has been more than eager to accept mobile technology as the mainstay of everyday life. Due to initial hesitation, the business world has been left to essentially play catch up (13). This game of catch up is not unknown by the business world. Research has shown that the “impact of emerging technologies on organizational processes has often been overestimated in the short term and underestimated in the long term (13).” The following text will attempt to describe the advantages that the adoption of mobile technology can bring to an organization, as well as the processes that are involved in such adoption.

2.3.1 Advantages to the Implementation of Mobile Technology

While a corporate organizations initial hesitation to accept new technology is understandable, companies are now beginning to understand that the utilization of such technology can provide an organization with significant benefits. Some advantages that the incorporation of mobile technology holds include:

- “Mobility: Users can conduct business anytime and anywhere;
- Flexibility: Data can be captured at the source, or point of origin, and ‘reverse wireless’ services can be provided, including problem area alerts...;
- Dissemination: Some wireless infrastructures support simultaneous delivery of data to mobile users within a specific geographical region. This functionality offers an efficient means of disseminating real-time information to a large user population, thus providing another avenue to enhance and improve customer service (14).”

Other benefits of mobile technology can include cost savings, and enhanced productivity (15). Additionally, increased profitability, while not directly relevant to a DOT, can be seen as an indicator of the ability of a mobile application to impact an industry. Research performed on the impact of Information Technology (IT) in industry found a significantly positive correlation between the implementation of IT resources and an organization’s profit ratios (16).

2.3.2 Fundamental Objectives in Developing a Mobile Solution

There are many different factors that play a role in the success and application of mobile technology within an organization. Researchers are able to enumerate these factors through the implementation of value based thinking, a process that incorporates brainstorming based upon values in an effort to identify superior decisions or alternatives (17). This process of value-focused thinking is presented in Figure 2.4.

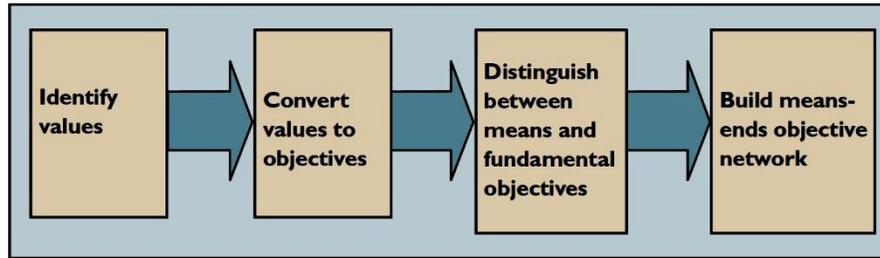


FIGURE 2.4 Procedures of Value-Focused Thinking (14)

For the specific case of mobile applications within industry, researchers have invoked the process of value-focused thinking using case studies to build a means-end diagram. This diagram, presented in Figure 2.5, describes the process of thinking that researchers underwent in order to produce an end goal or in other words to produce fundamental objectives that the incorporation of mobile technology will address. These fundamental objectives can be summarized six different statements: maximize customer satisfaction, maximize effectiveness, maximize efficiency, minimize cost, maximize employee acceptance, and maximize security. These statements can be further refined to reflect their core principles: efficiency, effectiveness, customer satisfaction, security, cost, and employee acceptance (14). It is these core principles that any application of mobile technology should attempt to satisfy.

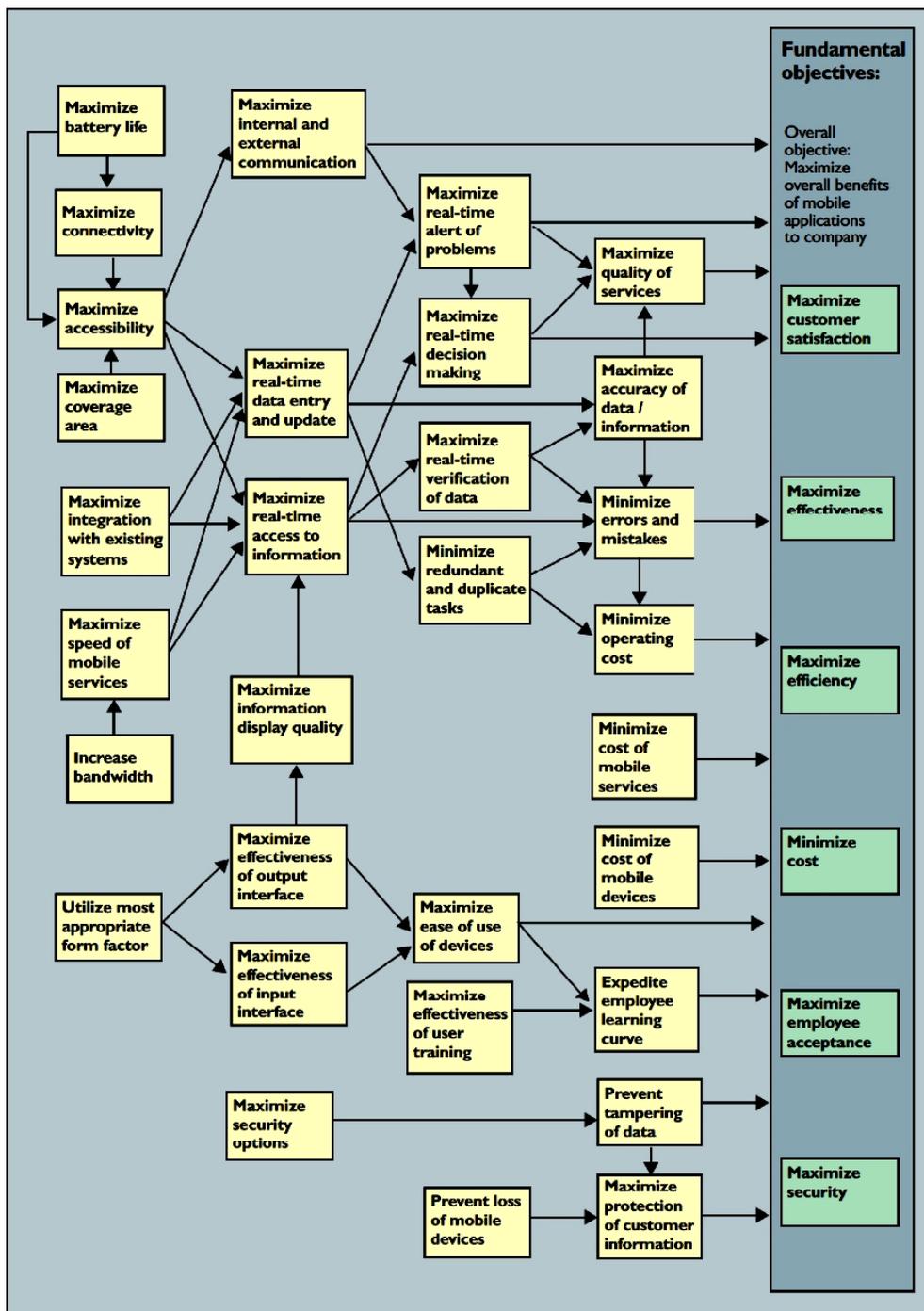


FIGURE 2.5 Means-Ends Objective Network for a Mobile Application (14)

2.3.3 Success Factors

Successful implementation of mobile technology depends on a number of different variables. Most of these variables can be grouped into two categories: Viability and Fit. The relationship between these categories is represented in Figure 2.6. In order to prove successful, the implemented applications “should be the ones that have a good fit between task and technology, and strong viability within the organization (18).” Applications that fail to provide both a high viability and high fit need to either be modified or abandoned for a solution that will provide a better fit.

Viability	High	Find Alternative Technology	Good Target
	Low	Forget it	Organizational Restructuring
		Low	High
		Fit	

FIGURE 2.6 Fit-Viability Framework (18) The category “Fit” can be better defined as how well the task can be addressed by the mobile solution. Five indicators, or attributes, that indicate a solution’s fitness are defined as:

- *Ubiquity*: Available at any location at any time;
- *Convenience*: Convenient for users to operate;
- *Instant connectivity*: Easily connected to the target;
- *Personalization*: Allows for preparation of personalized information;
- *Localization*: Location-specific information and products (19).

These attributes can be summarized for ease of use into three specific questions:

1. “Does the user need the product or service when on the move to different places?”
2. Will the value of the product or service decay substantially, or will the outcome will be disastrous, if users do not get the product or service on time?
3. Is the product or service different for different users (18)?”

The answer to these questions, when based on a scale, can be used to determine if a mobile solution will provide an adequate fit. “If a task’s requirements meet these criteria, its fit with mobile technology would be high (18).”

Viability refers to “the extent to which the infrastructure of the organization is ready for the application (20).” “In other words, viability assesses the fit between a mobile application and its associated users (18).” The viability of a mobile application comes from a combination of cost-benefit factors, organizational factors, and the current infrastructure/accessibility of technology to name a few (20).

While there are many factors that contribute to a mobile solution’s viability, one of the most important comes from a cost/benefit analysis. An application must be cost effective, both monetarily and in relation to a user’s transaction cost in order to be a viable option for an organization. In terms of finance, the following questions need to be addressed:

1. “Is the target application a frequent business process?”
2. Does the transaction process involve high uncertainty?
3. Is the current equipment and technological architecture adequate for supporting the application?
4. Is the cost for using wireless services high or low (18)?”

The financial costs of a mobile solution are not the only costs that need to be analyzed in a cost/benefit scenario. Perhaps superior to monetary cost is the need to minimize a users transaction costs in relation to the mobile solution. When considering transaction costs, the following questions need to be considered.

1. “How much time and effort are necessary for the user to learn the application?”
2. Does the application require a reallocation of resources (18)?”

These transaction costs may be minimized if the user population meets certain conditions. The following questions are meant to address those conditions in order to provide further guidance in assessing viability.

1. “Does top management support the application?”
2. Is the popularity of the intended mobile devices high?
3. Do people like to use the intended mobile device?
4. Are there popular substitute services available (18)?”

The questions above provide a guideline for assessing viability as a factor in the determination of a mobile application’s potential success. Upon completion of this questionnaire, an analyst should be able to determine if a mobile solution is viable or not. “If the

application is a good fit with the task and is highly viable in the organization or society, then it is likely to succeed (18).”

2.4 Process Evaluation

As mobile technology continues to evolve, the impact that it is having in the business world continues to grow. While initially hesitant, businesses are beginning to accept and find applications for mobile technology (Gebauer and Shaw 2004). Since this acceptance is relatively new, the research relating to determining the effect of mobile applications is somewhat limited. This section’s purpose is to present the available research on mobile process evaluation.

2.4.1 Usability

When considering the evaluation of a specific process, such as the application of a mobile solution, usability plays a pivotal role. Usability is the measure of the extent to which a user is able to use a product in an effective, efficient, and satisfactory manner. It needs to be considered in the design of a mobile application and then analyzed in the time following deployment of such applications (21).

ISO 9241-11 is an international standard that presents a definition of usability and an accompanying explanation of the process used to identify information necessary to evaluating the usability of a visual display terminal in terms of user performance and satisfaction (21). According to the standard, usability is the “extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use (21).”

To better understand this topic, definitions of the measures of usability need to be presented. Usability can be measured by effectiveness, efficiency, and satisfaction. Effectiveness is the “accuracy and completeness with which users achieve specified goals (21).” “Measures of effectiveness relate the goals or sub-goals of the user to the accuracy and completeness with which these goals can be achieved (21).” In other words, effectiveness “can be measured by the percent or number of errors, the success-failure ratio, or the percentage of tasks completed (22).”

Efficiency can be defined as a measure of the amount of “resources expended in relation to the accuracy and completeness with which users achieve goals (21).” “Measures of efficiency

include time taken to complete tasks, time spent using help or documentation, and estimations of subjective workload (22).” Other measures include mental or physical effort required, time commitment, and material or other financial cost (21).

The final measure of usability, satisfaction, is a user’s “freedom from discomfort and positive attitudes towards the use of the product (21).” “Measures of user satisfaction include the number of times the interface misleads the user or the user loses control of the system, the percent of favorable/unfavorable comments, and ratings on post-test questionnaires (22).”

2.4.2 Productivity

Productivity is another indicator within process management that is important to consider in an evaluation of the effectiveness of a mobile application. “Productivity can be defined simply as output divided by the input that is used to generate the output (22).” In past years there has “been considerable uncertainty and concern about the productivity impact of IT being experienced in work organizations (23).” Recent research, however, has quelled the concern and shown a positive correlation between IT and productivity given modern technology (24).

Productivity can be analyzed in a number of different ways, including total productivity, partial productivity, and physical productivity. Total productivity refers to the total output divided by the sum of all of the inputs. This measure of productivity is difficult to generate due to the different types and large quantities of inputs (22). In order to simplify productivity analysis, partial productivity measures are typically used as a proxy for total productivity. Partial productivity “is the ratio of gross or net output to single factor input (25).” While partial productivity is a simplification of total productivity, it can still be difficult to calculate due to the need for total output in calculations. Partial productivity is, however, much simpler than total productivity and is a widely used approach (25). In the case of not being able to determine total output, an even simpler alternative to total productivity can be implemented (22). This method, called physical productivity, is calculated by “dividing some typical output by an essential input (22).”

2.4.3 MoBiS-Q

There are a variety of tools available for measuring the usability and productivity of an application of IT. These tools, in the form of questionnaires, include System Usability Scale

(SUS), Software Usability measurement Inventory (SUMI), Questionnaire for User Interface Satisfaction (QUIS), The After-Scenario Questionnaire (ASQ), Post-Study System Usability Questionnaire (PSSUQ), and The Computer System Usability Questionnaire (CSUQ) (22). Each of these tools seeks to help a researcher determine the effect IT has on a user.

While extremely important for specific IT applications, these tools do not address the use of mobile services for business use (22). “The limitation of these questionnaires is that most of them were developed for a desktop environment and they do not cover the challenges in the usability of mobile applications (26).” These challenges include mobile context, connectivity, small screen size, different display resolutions, limited processing capability and power, and data entry issues such as small buttons and labels (27). Traditional IT analysis tools are also unsuitable for mobile application analysis because they “only address usability or productivity (22).” An appropriate analysis tool for analyzing the effect of mobile services would incorporate both of these topics (22).

To respond to the lack of analysis tools for mobile applications, researchers created a Mobile Business Service Questionnaire (MoBiS-Q). This tool presents an approach to analyzing the “level in which [users of a mobile] service perceive the usability and productivity of the service to be (22)” This method is loosely based in form upon the SUS questionnaire tool mentioned previously while its questions are based off of the psychometrically proven IT questionnaire tools previously mentioned.

After extensive case study analysis, ten themes were identified that would likely have a significant effect on the success of a mobile service (22). The themes include “ease of installation, learnability, ease of use, efficiency, effectiveness, user satisfaction, factors related to mobile context, safety, support, and mobile work productivity (26).” Each theme is accompanied by a number of questions, shown in Table 2.1, which are given to a mobile application user in conjunction with a Likert-scale. The user then assigns a rating to each question which can be used for analysis.

Upon collection of the rating data, analysis can be performed to determine the effectiveness of the mobile application. “The results of the MoBiS-Q can be used, for example, to analyze and benchmark the level in which the users of the service perceive the usability and productivity of the service to be. Moreover, by reusing the questionnaire, results may be used

for detecting changes, for example, in satisfaction and productivity over time (22).” This user-feedback type of analysis is an excellent method for determining usability and effectiveness. It should, however, be used in conjunction with objective performance data in order to substantiate the questionnaire’s results (22).

MoBiS-Q presents a viable option for measuring the success of mobile business services. It offers an effective way to collect a data set in a timely and accurate manner. Despite the relatively newness of the MoBiS-Q approach, it has already been successfully implemented in practice, further confirming its importance and validity. Pau Giner, et al. recently published work on the use of mobile devices in a smart business environment. Mobile devices were used to provide information and notifications based on location and proximity rather than user input. MoBiS-Q was implemented with a Likert scale in order to determine the usefulness of the mobile applications produced (28). With the increased implementation of mobile technology in the corporate world, MoBiS-Q will likely be increasingly implemented in future research.

TABLE 2.4 MoBiS-Q Themes and Questions with IT Survey Reference (22)

Theme	Questions
Installation	The mobile service was easy to assemble, install, and/or setup into the mobile device (QUIS, MPUQ) The speed of installation was reasonable
Learnability	It was easy to learn to use the mobile service (SUMI, QUIS, PSSUQ, MPUQ) Early experience with other mobile services (or products) made it easier to operate with this service
Ease of use	It is easy to move from one part of a task to another with the mobile service (SUMI, MPUQ) The organization of information in the mobile service is clear (SUMI) The mobile service has all those functions and capabilities that I expected it to have (PSSUQ) It is easy to navigate between hierarchical menus, ages, and inside of each screen with the mobile service (QUIS, MPUQ)
Efficiency	I can complete my work tasks quickly by using the service with a mobile device (PSSUQ) Some work tasks are too slow to complete with the mobile service The use of the mobile service in my work tasks requires a lot of mental effort The response times are fast enough in the mobile service (SUMI, QUIS)
Effectiveness	The work task sometimes fails because of the mobile service The mobile service enables quick, effective and economical performance of work tasks (PSSUQ)
User satisfaction	I would recommend the mobile service also for others doing the same work Using the mobile service in performing the work tasks is attractive and pleasing (SUMI) I feel confident using the mobile service (SUS)
Factors related to Mobile context	The battery capacity of the mobile device is sufficient for the use of the service for work tasks The screen size of the mobile device is adequate for using the service for work tasks Inputting information to the mobile device is easy The mobile device suits well for performing my work tasks Using the mobile device with one hand is easy Sometimes, the environment (such as coldness, sunshine, darkness) makes the use of the service difficult Switching between the different applications (e.g., calls and work task) in the mobile device is easy Exchanging and transmission of data between the mobile service and other products (e.g., computer, other mobile devices) is easy (SUMI, QUIS, MPUQ)
Safety	The use of the mobile service has caused me safety risks I sometimes have to fully concentrate on using the mobile service and cannot observe the environment Using the mobile service while on move is easy It is easy to perform work tasks in a hurry with the mobile service
Support	I always know who to ask for help if I have problems in performing work tasks with the mobile service or device The help information given by the mobile service is useful (SUMI) The mobile service gives all the necessary information for me to use it properly in my work tasks
Mobile work productivity	Using the mobile service in my job reduces travelling from and to the office during the work day. I can complete my work tasks in less time than before due to the mobile service. Using the mobile service in my job increases my productivity. (TAM) Using the mobile service in my job increases my work motivation.

2.4.4 Likert Scales

The questions invoked by MoBiS-Q are best posed to a mobile user in conjunction with a Likert scale. This type of scale is a very common rating system that is used on surveys. Because of this scales importance to in evaluation, a brief discussion and research review is presented in the following paragraphs.

“As individuals attempt to quantify constructs which are not directly measurable they oftentimes use multiple-item scales and summated ratings to quantify the construct(s) of interest (29).” This type of rating involves the use of a Likert scale. The Likert scale is best described in the following manner:

A set of items, composed of approximately an equal number of favorable and unfavorable statements concerning the attitude object, is given to a group of subjects. They are asked to respond to each statement in terms of their own degree of agreement or disagreement. Typically, they are instructed to select one of five responses: strongly agree, agree, undecided, disagree, or strongly disagree. The specific responses to the items are combined so that individuals with the most favorable attitudes will have the highest scores while individuals with the least favorable (or unfavorable) attitudes will have the lowest scores (30).

In using a Likert scale, care must be taken in interpreting the results. “The data collected are ordinal: they have an inherent order or sequence, but one cannot assume that the respondent means that the difference between agreeing and strongly agreeing is the same as between agreeing and being undecided (31).” The data can be described using either a median or a mode (ideal), but not a mean due to the nature of the data. Variability should be described using inter-quartile-range rather than standard deviation. Also, the data should be displayed on a bar-chart type graph rather than a histogram due to its discontinuous nature (31). Additionally, “nonparametric procedures—based on the rank, median or range—are appropriate for analyzing these data, as are distribution free methods such as tabulations, frequencies, contingency tables and chi-squared statistics (32).”

A point of concern in using Likert scales is where a respondent is undecided, or midpoint. Research performed in this area, however, indicates that the percentages of undecided responses are not crucial in the analysis of Likert scale data. “Significance tests produced the same results whether "don't know" responses were included or excluded for several items that had high percentages of "don't know" responses, and for several items that had low percentages of those

responses. Furthermore, significance tests produced different results from one condition to the other for items with large percentages of "don't know" responses and for items with small percentages of those responses (33)." The validity of the Likert scale, therefore, remains, providing a valuable tool for simple and informative analysis of attitude.

2.4.5 Survey Creation

Creation of a workplace survey needs to be approached with an understanding of workplace dynamics and sensitivity to the need for anonymity in user response. These considerations are addressed within the topics of survey length and administration method as presented in the subheadings below.

2.4.5.1 Optimal Survey Length

According to a study regarding the "Effects of Questionnaire Length on Response Quality" performed by Herzog and Bachman (34), even lengthy questionnaires can be administered without a substantial depreciation in result quality. This can be minimized even more when efforts are made by survey administrators to maintain the motivation of respondents. This trend is supported in a separate study performed using a number of different survey administration methods (35).

2.4.5.2 Survey Administration Method

While there are many options available to facilitate the administration of a questionnaire, two feasible options exist that effectively maintain user anonymity. These methods are both self administered and can be offered either through traditional pen-and-paper administration or on a computer. In an analysis of such surveys, researchers determined that, assuming a familiarization with the technology through which surveys are administered, survey results are similar regardless of the administration method or the sensitivity of the survey data (36).

2.5 Conclusion

This review has discussed in detail research performed in the areas of transportation sign management, the development and implementation of mobile technology, and mobile technology process evaluation. This information will be built upon in the following chapters in an attempt to document the development, implementation, and analysis of a mobile solution within UDOT.

CHAPTER 3 APPLICATION DEVELOPMENT

3.1 Introduction

The purpose of this research is to develop a technology solution on which an analysis can be performed in order to determine the effect that incorporating mobile technology into a state Department of Transportation (DOT) will have. This chapter outlines the process of developing such a solution, the resources used, and approaches taken to ensure cross compatibility and usefulness. Detailed user's guides as well as programming documentation can be found in the Appendices.

3.2 Maintenance Shed Application

The first step in the development process for the maintenance shed application was to identify the key concerns that the application should address. These concerns, detailed in Chapter 1, can be summarized simply as the need for a living directory of information relating to the people, location, and equipment found within UDOT maintenance shed locations.

Once specific concerns were identified, a number of brainstorming iterations were taken in order to come up with the best process of retrieving the data. This resulted in a number of programming steps that would need to be taken in order to function efficiently. These steps were grouped into two different specific categories: database steps and application steps.

Database Steps:

- Obtain initial output (static) from UDOT Database
- Develop required Structured Query Language (SQLite) framework. This requires running the existing database through extensive splitting and modifications to output usable data instances rather than strings.
- Build local (or cloud) SQLite Server
- Push framework to Server
- Access UDOT database server for Read Only pull
- Set up SQLite Server to auto-populate with changes

- Cloud server may be the best way to do this as the application does not require a lot of storage space
- Debug

Application Steps:

- Probe recent needs and ensure app will address them
- Create list of all needed information
- Develop graphical user interface (GUI)
- Program application for Android and link to SQLITE Database
- Possible transfer to Apple iOS can be looked into
- Debug
- Field test
- This will likely occur more than one time in the process
- Beta test with selected UDOT employees
- Product Modifications
- Delivery
- Training

In addition to the development of brainstorming steps, the application was laid out in a mind-map to show the application's screen progression through the data retrieval process. This map, shown in Figure 3.1, was pivotal in helping to envision this progression. As a final component of the envisioning process, a preliminary name of ShedVIEW was given to the application.

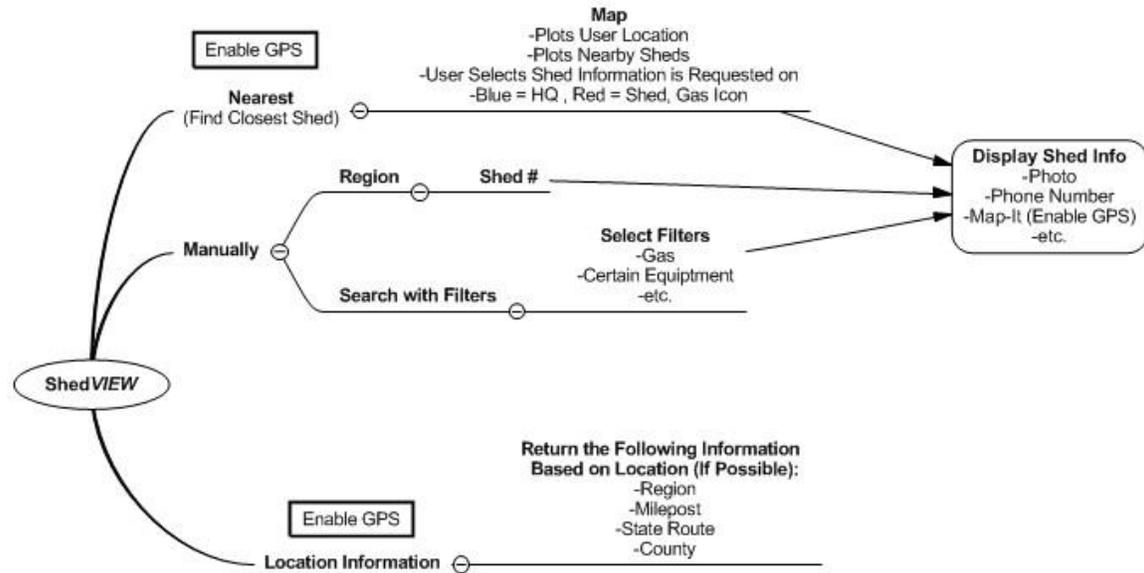


FIGURE 3.1 ShedVIEW Mind-Map

3.2.1 ShedVIEW Programming Efforts

Initially development of the ShedVIEW application was attempted in Adobe Dreamweaver CS5.5. Recent improvements to this program allowed a simple mobile query application to be created in HyperText Markup Language (HTML) and then be pushed to both iOS and Android mobile operating systems. Initial attempts in this program, however, did not prove fruitful. A trial application was produced during testing and it was found that the application was too slow and buggy to use in production. In addition to the buggy application, another problem that was not addressed by Dreamweaver was permission control. In general, a simple query application should not require a lot of information from the phone or about the user. For an undetermined reason, Dreamweaver automatically added in a large number of permission requests that were not needed, not to mention extremely intrusive. A number of unsuccessful attempts were taken to remove the permissions. A final issue faced in Dreamweaver programming was the fact that mobile development integration was new to CS5.5, Dreamweaver provided limited support and information regarding the development of applications in their program. More importantly, the tutorials that were available only described the development process and not the fairly complex production/distribution process.

Great lengths were taken to troubleshoot, simplify, and enhance the Dreamweaver test application to no avail. In further research of IT threads and discussion boards, it was found that the problem did not lie solely in the case of the attempted development of the ShedVIEW application, but rather in the approach that was taken by the Dreamweaver program in converting the programming from HTML to the mobile native Java needed to push an application to a device. Since the preliminary Dreamweaver development, a new version of the program (CS6) has come out which seems to promise a much better and intuitive system for mobile development. This addition came too late to be tested, however, and alternatives needed to be researched.

Because Dreamweaver trials resulted in a non-acceptable programming solution, additional attempts were made to find a program that would allow for the simple development of a mobile application. A number of alternate solutions were researched and tested but none of them led to a comprehensive development tool. One of these alternatives, MIT App Inventor was promising early on and was tested in the same manner as Dreamweaver. This app development tool is geared for the non-programming community and is excellent for simple applications that require limited functionality. After testing, however, it was found that there is limited control that a user of this tool can have in regards to the layout of the application. This was problematic for ShedVIEW due to the need for a more corporate themed application available in multiple screen size formats. Additionally, the programming requirements of ShedVIEW pushed beyond the limits of what could be feasibly accomplished using an App Inventor solution. Because of these two reasons and a number of additional minor functionality issues, It was determined that MIT App Inventor was not best suited for use in the development of ShedVIEW.

With the unsuccessful testing of simplified mobile development solutions, a comprehensive development tool was turned to in order to have complete control of ShedVIEW from start to finish. This tool, an Integrated Development Environment (IDE) called Eclipse, is meant in part to facilitate development of Android mobile applications in their Android-native Java programming language. Eclipse is the development tool of choice for professional programmers who require complete control over the application development process.

Eclipse by itself, however, is not a complete tool. In order to be used for Android development, an in-software toolkit has to be installed. Google has provided this set of tools and they can easily be integrated directly into the Eclipse IDE. This toolset, called the Android Development Tools (ADT) “is a set of components (plug-ins) which extend the Eclipse IDE with Android development capabilities. The ADT contains all required functionalities to create, compile, debug and deploy Android applications from the Eclipse IDE and from the command line. [The ADT] also provide an Android device emulator, so that Android applications can be tested without a real Android phone (37).”

Development of the ShedVIEW application within Eclipse proceeded initially in three different phases which correspond to three different ways of obtaining information. The first two ways offer a manual or GPS driven automatic search in order to return information specific to UDOT maintenance sheds. The third method returns information relating to the user’s location. The development of these methods will be described in detail in the following paragraphs. More specific programming notes and documentation can be found in Appendix C and Appendix D.

3.2.1.1 Manual Search

In an effort to better aid UDOT personnel in the access of information relating to different maintenance sheds throughout the state, development of ShedVIEW started with the objective of creating a method to manually filter through the four UDOT regions. The purpose of this filtering is two-fold. First, it allows a user to find information about a specific shed located in any part of the state of Utah. Second, this functionality allows an employee to obtain travel directions through the built in system navigation application for trips to off-site shed locations. The incorporation of a manual filtration process into the application means an employee does not need to be nearby the maintenance shed of interest in order to find information about it.

The method in which the information filters are presented to the user is fairly straightforward. Upon initiation of the ShedVIEW application, the user is presented with a screen containing two easy to view buttons, shown in Figure 3.2. These buttons present the different ways to obtain information. Once the user selects the manual search button, a separate

screen is visible that asks if the user would like to search by UDOT region or by the city a shed is located in. Upon selection of the UDOT region button, the user is presented with each of the four UDOT regions. Selection of a region will then bring up a list of all the sheds along with the city that they are located in that are available within that region, as represented in Figure 3.3. The user then selects a shed and is presented with different information including management personnel contact information, employee directories, and equipment lists. This information is populated directly from the SQLite database. The user is also presented with a button to call the station, and a separate button to map the station location and obtain driving directions based based on the user's current location.

Alternately, a user can select to search by the cities in which a maintenance shed resides. This was incorporated into the application upon the determination that employees may know the city that they want to visit more readily than a region and shed number. Once a user selects the option to search by cities, a new screen is visible which presents a search box along with a list of Utah cities which contain maintenance stations. The user can simply type in the city that is being searched for and make a selection. While the user is typing, the list of cities is limited, allowing for a quicker selection without having to type in all the characters of a city's name. Once selection of a city is made, the stations that are within that city are presented to the user for selection. Selection of a station brings up the same shed information screen that was discussed previously.

The purpose of the manual search functionality is to provide desired information to a user with minimal effort. In order to do this, care was taken in the programming initiative to ensure that the speed of the ShedVIEW application would not be hindered by complex processing procedures. With the processing power of current mobile devices, it is to be assumed that if care is taken in the programming phase, the non-complex retrieval procedure that ShedVIEW implements will result in a non user-laborious information retrieval process.

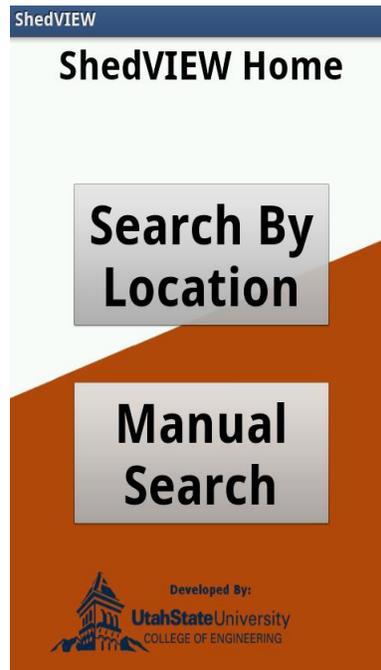


FIGURE 3.2 ShedVIEW Home Screen



FIGURE 3.3 ShedVIEW Search by Region *Automatic Search*

In an effort to make the retrieval of information relating to maintenance sheds even easier, an automatic search function is built into the ShedVIEW application. The main purpose of this application is to provide UDOT personnel who are out in the field with a way to determine their proximity to nearby shed locations. The information that is presented at the end of the search is exactly the same as is presented in at the end of the manual search process, the difference lies in how the filtering process is presented to the user.

Upon selection of the automatic search button option from the ShedVIEW home screen, the user is presented with an operating system (OS) default sub-screen which asks the user to enable the device GPS if it is not already enabled. Once enabled, the application is programmed to pinpoint the user's current position and find maintenance sheds that are nearby. There are two different ways that a user can view the results of this GPS driven search. Results can be presented in a map view or a list view.

The first viewing method is presented in a Google-maps driven window that shows the user's location in a balloon along with nearby shed locations in balloons that are colored different than the location balloon. From this display, a user can zoom in to find a desired location and select the balloon to pull up the shed information mentioned previously. This map-

driven presentation is a very quick and easy method for determining proximity to nearby maintenance sheds.

An alternative to the map view can be selected by the mobile user for clarity and ease of use. This method is still driven by GPS, but rather than presenting shed location information on a map, it displays a number of sheds that are nearby, along with the user's distance to each shed. The user can then pick the shed which is of most interest and then directly access shed-specific information. This method is also very fast and is an excellent option for a user who is looking for a specific shed that is nearby.

In the process of developing the automatic search functionality, it was determined that certain maintenance sheds provide certain services that other shed locations may not offer. In both the map view and the list view of the automatic search, a provision was made to represent desirable services, such as gas station accessibility directly within the search results. This is meant to provide clarity to the mobile user, as well as eliminate the need for iterations in order to find desirable information.

3.2.1.3 Location Specific Information

A third option that is meant for inclusion in future releases presents a user with a location-specific information button on the home screen. While this option is programmed using the same principals as the manual and automatic search methods, the functionality is completely different. This module is meant to be built on as the application continues to build support and gain use among UDOT employees. It is designed with the idea that any geographical information that needs to be determined can be presented to a user given the correct programming infrastructure is developed.

Initially, this option will provide some less programming-intensive information relating to the user's current location. Upon selection of the Location-Specific Information button, the user is presented with the option to enable GPS, as was presented in the automatic search functionality. Once enabled, the user location will be determined and then will return the current State County, and UDOT Region in which the user is located. This will be returned to the user in a listed out manner on a new blank screen.

Once the ShedVIEW application gains ground and acceptance among UDOT employees, there is a possibility for a number of additional information fields to be returned to the user. These fields could include the specific milepost and State Route that a user is located on. They could also include the user's distance to any of a number of locations throughout the state of Utah. Should the need arise for additional information to be returned by this functionality, the provision for additional programming and implementation can occur within this search option.

3.2.2 Database Programming Effort

Android by default runs off of a lightweight database called SQLite. Unlike its parent version Structured Query Language (SQL), SQLite occupies a small amount of both disk space and memory, two resources that are valuable in mobile computing. "Rather than running independently as a standalone process, it symbiotically coexists inside the application it serves—within its process space. Its code is intertwined, or embedded, as a part of the program that hosts it (38)." Additionally, because SQLite runs on the public domain, a developer is free to do anything with the program without any type of licensing restriction (38).

A number of GUI tools exist for the development of a SQLite database. After researching a number of different options, SQLite Manager, an add-in for Mozilla Firefox, was chosen as the best choice. This program allows for the importing of a Comma-Separated Values (CSV) data table directly into SQLite format. From this database, queries can be run by the Android operating system to return specific information.

The database was originally programmed using current output from the UDOT OMS. The output was saved in a Microsoft Excel file and then imported directly to the SQLite database manager. Once read-only privileges are set up within the UDOT OMS, this data can be automatically transferred directly into a SQLite database and then saved in a way that the application can receive the updated information.

The key factor in the success of ShedVIEW is being able to access up to date information in locations where internet access may not be readily available. This is made possible through the implementation of a data server. While programming efforts are ongoing the process that the server will invoke has been researched and determined. The server will be programmed to pull information as it changes in UDOT servers, and then to convert that information into a useable

SQLite database. Once the database is populated, the server will send out a notification update to the mobile devices that are linked to the server. Assuming internet connectivity, the mobile device user will be presented with an option to either accept or reject the update package. Upon acceptance, the mobile device will download the updated database, overwriting the old database that was on the phone originally. This ensures that a living database is always accessible to a mobile user. Additionally, by downloading the database directly onto the mobile device, speed is greatly increased and the user is able to access directory information regardless of cellular connectivity.

3.3 Sign Management App

The sign management application development greatly differed from ShedVIEW in that the programming did not start from the ground up. Rather than start from scratch, this application is a special application of the open-sourced survey mobile application system called Open Data Kit (ODK). “ODK is a free and open-source set of tools which help organizations author, field, and manage mobile data collection solutions (39).” Originally developed by researchers at the University of Washington in an effort to explore “how technology can improve the lives of under-served populations around the world (39),” ODK’s value is being realized through a number of industries where data acquisition is commonplace.

There are three main components of ODK that are used to create a data collection survey, collect data given the created survey, and then to gather the collected data for use. These components are named as XLSForm, ODK Collect, and ODK Aggregate respectively. Because each function in a different way to accomplish the purpose of data collection, they will be discussed separately in the following paragraphs starting with ODK Collect.

3.3.1 ODK Collect

ODK Collect is the GUI of the ODK system. It runs only on Google’s Android operating system and has been developed in an open-sourced coding environment. Its purpose is to present data in a highly professional and intuitive manner, thus striving to limit user bias and false input. According to the developers, “ODK Collect renders forms into a sequence of input prompts that apply form logic, entry constraints, and repeating sub-structures. Users work through the prompts and can save the submission at any point. Finalized submissions can be sent to (and

new forms downloaded from) a server. Currently, ODK Collect uses the Android platform, supports a wide variety of prompts (text, number, location, multimedia, barcodes), and works well [with or] without network connectivity (40).”

ODK Collect offers the versatility of Android in that it can be installed either from Google’s App Market called Google Play or it can be installed from a file circulated over any file distribution method. Once installed, the program acts as an empty shell and needs to be filled with a survey questionnaire in order to be of any use. Once a questionnaire is installed, however, the application is able to present data collection topics in a professional looking fashion that is great for viewing in both low and high light situations.

Navigation through the application is accomplished by swiping a finger from right to left to advance or from left to right to return to a previously addressed topic. In the case of finger inaccessibility due to cold weather gloves or other protective hand coverings, navigation can still take place using a capacitive stylus in place of a finger. Upon navigation to the end of a questionnaire, ODK Collect will prompt the user to save and close the survey. This action saves the survey along with any accompanying media to the local mobile device. Upon completion of data collection activities, results can be uploaded over an internet connection to ODK Aggregate for further manipulation.

Because ODK Collect is purely a shell that fills a means-ends purpose, a better understanding of its functionality will come from learning more about what is being fed into the application in terms of a questionnaire and also where the information goes once it leaves the application. ODK Collect will be revisited in the following paragraphs in more specific context as it relates to facilitating the functionality of both XLSForm and ODK Aggregate.

3.3.2 XLSForm

XLSForm is a tool that allows an administrator to input questions and appropriate response fields into a Microsoft Excel worksheet. Once the worksheet is complete, XLSForm provides a free tool to convert the expressions into a required Extensible Markup Language (XML) format (41). This process of Programming in Excel and then converting to XML allows for a much more intuitive and less labor intensive questionnaire creation. It is also much easier

to perform change and improvement iterations on the questionnaire in Excel. Once a change is made, the form is simply converted again to XML and can be input directly onto the device.

XLSForm takes advantage of a number of widgets that are built into the Android operating system in order to create a user friendly input form. These widgets include but are not limited to such things as drop down menus, button selection fields, date pickers, and text input. Users can input data into the application in a number of different ways, as shown in Table 3.1. In addition to user-input data, some forms of data can be collected automatically when specified within the XLSForm programming. These data fields, called metadata, are shown in Table 3.2, and are mostly specific to the device being used to record data. When implemented into ODK Collect, questions are prompted for user input while metadata is simply recorded in background tasks without the knowledge of the user. This background information is useful in identifying common errors across a dataset, as well as providing date information automatically.

TABLE 3.1 ODK Question Types (42)

Text	Text input.
Integer	Integer (i.e. whole number) input.
Decimal	Decimal input.
Select One	Multiple choice question; only one answer can be selected.
Select Multiple	Multiple choice question; multiple answers can be selected.
Note	Display a note on the screen, takes no input.
Geo-Point	Collect GPS coordinates.
Image	Take a photograph.
Barcode	Scan a barcode, requires the barcode scanner app is installed.
Date	Date input.
Date-Time	Accepts a date and a time input.
Audio	Take an audio recording.
Video	Take a video recording.
Calculate	Perform a calculation; see “calculates” below.

TABLE 3.2 Metadata (42)

Start	Start date and time of the survey.
End	End date and time of the survey.
Today	Day of the survey.
Device ID	IMEI (International Mobile Equipment Identity)
Subscriber ID	IMSI (International Mobile Subscriber Identity)
IMEI	SIM serial number.
Phone Number	Phone number (if available).

The questionnaire for sign survey was built within XLSForm in part from fields already present in UDOT's OMS. The OMS contains two tables that store different information about each sign recorded. The first table contains information about the sign supports while the second table contains information relating to the sign faces themselves. Having two tables enables UDOT to be able to identify supports that have more than one sign on them. It also helps to keep the database in an organized manner. A description of each of the fields required by the OMS is provided in Appendix E.

Aside from some minor problems with the way questions are presented, along with a few erroneous responses, the current OMS provides a wealth of information. This information can be improved, however, with the slight modification of a small number of the fields in both the sign and support sections. These changes are grouped for clarity into two categories: deletions and additions.

In order to facilitate data entry, it is suggested that a few question fields be removed entirely from the OMS. Under the support section, these questions include the start milepost, and the x and y coordinates. If a move to a GPS driven database is to be made, these questions become obsolete. In the sign section, it is suggested that only a single question asking a maintenance worker to enter information regarding the sign legend text be removed from the OMS. This instance of data is an unneeded burden due to the implementation of a required photograph data field that will tie into the database. The removal of this field will speed up the data entry process, as well as remove the chance for error in typing a text-based response.

With the implementation of mobile devices in data collection, a number of information fields need to be added to the OMS in order to better represent both the data and the device recording the data. For both the sign and support sections of the OMS, it is suggested to add fields specific to the device identification (ID), and the device phone number. This information will aid investigation in the case of incorrectly entered data. It can also help determine if a device is in need of an updated data collection form. Additionally, fields for a barcode, picture, and geo-location need to be inserted into the OMS to provide information fields that replace those that were suggested for removal above. Also, in order to facilitate the flow through survey fields, a field for selecting the maintenance performed needs to be included in the OMS. Finally, the OMS would benefit from information regarding the installation date of the sign or post, as well as its overall pass/fail status. These will aid UDOT personnel in analysis and monitoring population related issues.

In addition to these suggested information fields, there are three fields that are suggested for inclusion specifically in the sign portion of the OMS. Two of these fields relate to damage, both major and minor. The options that would be presented to a user for each selection would include none, bending, peeling, vandalism, cracking, fading, and other. The third field that is suggested for inclusion is retroreflectivity data which has been mentioned previously as an important aspect of a SMS. Each of these fields will provide an additional understanding of the true status of the sign and its ability to convey its message.

With the aforementioned addition and deletion of these OMS field instances, each of the suggested inclusions presented in Tables 2.1, 2.2, and 2.3 will be accounted for. This information will provide UDOT with tools to meet planning and accountability requirements and facilitate effective sign population management.

3.3.3 ODK Aggregate

Once data is collected within the ODK Collect GUI, the information is pushed from the device to a server that has been set up for the collection of data from multiple devices. This server, called ODK Aggregate, has many different functions. It's main function, however, is to act as a data repository (43). The detailed functions of ODK Aggregate that are applicable to the

Sign Management application are the ability to accept submissions from ODK Collect, manage collected data, and the ability to export data into CSV files for use in Microsoft Excel.

For small scale operations, ODK Aggregate can be deployed to Google's cloud servers which then provide a location for the data to reside once pulled from mobile devices. Aggregate is built in such a way as to provide seamless integration into the cloud, therefore installation of the server to the cloud is fairly straightforward. There are limits, however, in the usefulness of the Google Cloud Service. In considering alternatives to Google's service, the ODK developers offer the following issues to consider on the development web site (44):

Internet access — Google App Engine requires internet access. If you don't have internet access, consider using only ODK Briefcase; it may be more appropriate. Tomcat deployments can operate without internet access; in such an environment, ODK Collect would only be able to upload finalized forms after it connects to the network containing the Tomcat deployment.

Computer skills — Tomcat deployments have a steep learning curve and require technical aptitude. At a minimum you will be (1) changing network configuration, (2) either selecting and using a website hosting service or specifying and configuring your own server and network router(s), (3) installing software, and (4) ensuring that your site has proper power-failure and data-backup systems in place.

Ongoing support — Tomcat deployments require periodic backups of your data; if data security is a concern, you should have a part-time system- and/or database-administrator to periodically review logs and look for malicious activity.

Availability — Most users provision ODK Collect with blank forms and rarely update those forms over the course of a study; surveyors upload finalized forms to ODK Aggregate infrequently and opportunistically. If that is your situation, you likely do not need as high-availability a server as Google App Engine provides. Google App Engine cloud services provide highly available servers and data storage. Tomcat deployments with similar availability will be expensive to operate unless your organization already has its own information technology department. The less downtime you can tolerate, the more expensive a Tomcat deployment will be.

Dataset size — Google App Engine can store an unlimited amount of data. However, for large datasets (over 7,000 records), the "Export to CSV" and "Export to KML" functions will cease to operate unless you perform a custom deployment to increase the size of the server on which the background processes run (this also applies to Apache Tomcat installations). On Google App Engine, this larger server will incur higher billing costs. Additionally, for very large

datasets (over 100,000 records), it is highly likely that performance will be better when using MySQL or PostgreSQL, and it is certainly true that you have more optimization opportunities when running your own database servers than are available through Google's cloud services.

Data locality and security — Google App Engine servers may be located anywhere in the world. Depending on the sensitivity of the data and specific storage rules/restrictions of your country or organization, the server infrastructure may not have all necessary locality guarantees or security precautions. In some circumstances, you might be able to use Encrypted Forms to achieve compliance. You should research and comply with applicable laws and regulations before storing data on Google App Engine. See the Google App Engine Terms of Service.

Billing — Google App Engine has 24-hour activity quotas that typically enable free use of ODK Aggregate during evaluation and small pilot studies. You may be able to run a full study within these activity thresholds provided you are collecting fewer than 2000 responses, access the site only during the work day (not 24hr/day), and can be flexible as to when you upload data from ODK Collect or access that data (should the quota have been exceeded). Otherwise, you will need to set up a billing account with Google.

In order to best meet the long-term needs of UDOT, the limits of Google's Cloud Service are unacceptable. The alternative to this service is to set up a local server instance that can be controlled and secured. This is reasonably accomplished through the use of an open source server software called Apache Tomcat. This high-powered server software has been downloaded over ten million times and is used by large companies such as Wal-Mart, E*Trade, and The Weather Channel (45). The developers of ODK offer Tomcat server installation procedures, located on the ODK website, which while technical, are straightforward.

Once the server is set up and ODK aggregate installed, Aggregate functions in the same basic way that it does on the Google Cloud Service. Submissions are sent from a mobile device to the Aggregate server. Aggregate collects the submissions and stores them on the local hardware. This service brings together submissions from many different collection devices into one location. Once on the local aggregate server, the information sits until a scheduled dump is performed that exports the data instances into a CSV file that can then be imported into a sign management system. The actual process of exporting and then importing into a SMS is beyond

the scope of this research. Additional research needs to occur in order to best optimize and automate that process.

3.4 Conclusion

Great lengths have been taken to ensure that each of the mobile applications built for UDOT were developed in a way that facilitates productivity and efficiency. ShedVIEW has been programmed in an attempt to solve the living information problem in regards to maintenance shed information. In an effort to ensure greater usability, the programming effort has gone above and beyond maintenance shed information, providing useful information that can be used by a UDOT maintenance worker for proximity and location confidence. A second application was developed in a manner to facilitate data collection as part of a traffic sign management system. This application offers an intuitive and seamless flow of information from maintenance work in the field directly to a management database.

In the future, both of these applications will be implemented within the UDOT division. This implementation will then be analyzed using the surveys and evaluation framework presented Chapter 4. This will provide understanding in an effort to determine the effect that incorporating mobile technology into a state Department of Transportation (DOT) will have. It is anticipated that the implementation of each of these applications will have an extremely positive affect within the agency.

CHAPTER 4 EVALUATION FRAMEWORK

4.1 Introduction

The intent of this chapter is to provide a detailed evaluation plan that can be used in future research in determining the effect that incorporating mobile technology will have within a transportation agency. To facilitate this determination, both user and developer surveys will be introduced using indicators outlined in Figure 4.1 in order to solicit responses regarding both the ShedVIEW and Data Collection Applications. These surveys will provide data on the mobile technologies usability, viability, and fit within the agency as well as their effect on user productivity. Finally, this chapter will include a framework that can be referenced in post-survey data analysis. Given this framework, it is anticipated that future research efforts will be able to evaluate mobile technologies effect within UDOT.

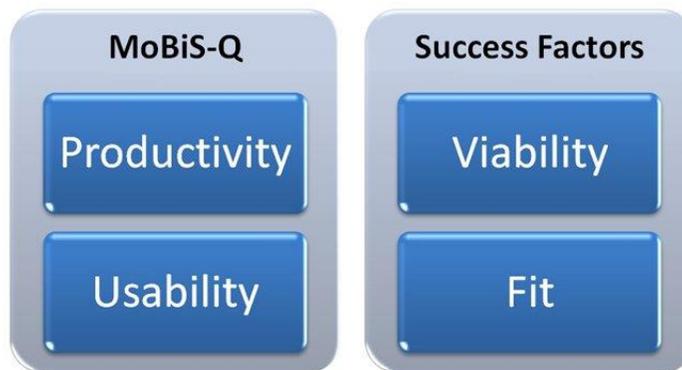


FIGURE 4.1 Effectiveness Indicators Sample Groups

UDOT is a large agency whose employees come from a variety of backgrounds and education levels. Because of this, it is theorized that the mobile application's implementation effects will differ based upon user demographics. In an effort to identify these differences, employees receiving the applications will be split into three groups according to employment position. These groups will consist of managers, engineers, and field personnel. It is anticipated that all users of the two applications developed will fall within one of these categories. Additional categories, created from age groups and education levels will be used in later sections of this research for further analysis.

4.2.1 Managers

The smallest sample group will consist of managers. These UDOT employees hold upper level positions that oversee various engineering and field activities. They typically hold educational qualifications superior to other employees and are on average, older and more experienced, possibly leading to reluctance in adopting mobile technology for everyday work practices. The ShedVIEW application will be the most used throughout this group of employees.

4.2.2 Engineers

The second sample group of UDOT employees consists of engineers. This group typically works both in the field collecting data and in an office environment. The majority of their time however is spent at the latter location. This group should be familiar with both of the developed applications but will primarily use the ShedVIEW application. This group is generally well educated and from a younger generation. It is therefore anticipated that this group will be the most accepting of the applications out of all three groups that will be analyzed.

4.2.3 Field Personnel

Of the mobile applications developed, the data collection application will see the most use from UDOT field employees. This group of employees will be the easiest and largest group to sample and will typically be employed at the Maintenance Shed level. They will use both of the developed applications on a daily basis to accomplish work in the field. It is also anticipated that this group of employees may exhibit a certain amount of bias due to a reluctance to modify practices that have become second nature. This reluctance will be addressed through the survey questionnaire.

4.3 Survey Design

In order to fulfill the need to determine the fit, viability, usability, and effect on productivity that a mobile application has within an agency, a questionnaire was developed. The following sections discuss the different components of the survey and then present a final paper-based survey for either direct use or for programming into a computer for a web-based survey approach.

4.3.1 Survey Development

In order to address each of the four indicators, namely fit, viability, usability, and productivity, two sources of questions need to be addressed. The first source of questions addresses the topics of productivity and usability and comes in the form of the pre-defined MoBiS-Q survey. Although this survey contains pre-defined question statements, the survey needed to be tailored to reflect the traits and features of the specific mobile applications. Because of this customization requirement two separate surveys were developed, one for the ShedVIEW application and another for the data collection application. The second source of questions measures the success factors of viability and fit. These statements did not need to be defined separately and were included in each of the final two user surveys. Because these statements were not pre-defined, they were carefully developed to address key indicators that are used to describe both fit and viability. These key indicators are described in detail in the following sections. Further questions specifically regarding viability were defined in a third survey designed for internal developer analysis rather than to be assessed on a user-level.

4.3.1.1 *MoBiS-Q for ShedVIEW*

The ShedVIEW application will be installed by a user on a personal device and is meant for personal use in the accomplishment of various work activities. As such, only some of the MoBiS-Q survey components were deemed as relevant to the purposes of this study. Reduction of the survey resulted in eleven statements, presented in Table 4.1, that address the various aspects of mobile implementation related to productivity and usability. Of these eleven questions, six address the topic of usability and five examine the topic of productivity.

TABLE 4.1 MoBiS-Q for ShedVIEW[Adapted from (22)]

Indicator	Theme	Statements
Usability	Installation	The mobile service was easy to install onto the mobile device
	Learnability	Early experience with other mobile services (or products) made it easier to operate with this service
	Ease of use	The organization of information in the mobile service is clear It is easy to navigate between hierarchical menus, ages, and inside of each screen with the mobile service
	Efficiency	The response times are fast enough in the mobile service
Productivity	Effectiveness	The mobile service enables quick, effective and economical performance of work tasks
	User satisfaction	I would recommend the mobile service also for others doing the same work
	Support	I always know where to look for help if I have problems in performing work tasks with the mobile service or device
	Mobile work productivity	Using the mobile service in my job reduces travelling from and to the office during the workday Using the mobile service in my job increases my productivity

4.3.1.2 MoBiS-Q for the Data Collection App

The Data Collection Application greatly differs from the ShedVIEW application and as such requires consideration of factors not addressed through the previously developed MoBiS-Q questionnaire. This application is meant to run on Tablet devices that are not user-owned and will be used in the field to accomplish specific work activities. As such, in order to provision for all possibilities it is assumed that the user has limited experience using such devices. This assumption is acceptable to make because survey information is only added and not lost from its provision. Given these qualifications, fourteen statements, presented in presented in Table 4.2, were selected from MoBiS-Q for inclusion in the Data Collection App survey. Of these, six address the topic of usability and eight explore the topic of productivity.

TABLE 4.2 MoBiS-Q for the Data Collection App [Adapted from (22)]

Indicator	Theme	Statements
Usability	Efficiency	I can complete my work tasks quickly by using the service with a mobile device Some work tasks are too slow to complete with the mobile service
	Factors related to Mobile context	The battery capacity of the mobile device is sufficient for the use of the service for work tasks The screen size of the mobile device is adequate for using the service for work tasks Inputting information to the mobile device is easy Sometimes, the environment (such as coldness, sunshine, darkness) makes the use of the service difficult
Productivity	Effectiveness	The work task sometimes fails because of the mobile service The mobile service enables quick, effective and economical performance of work tasks
	User satisfaction	I would recommend the mobile service also for others doing the same work
	Safety	The use of the mobile service has caused me safety risks
	Support	I always know who to ask for help if I have problems in performing work tasks with the mobile service or device
	Mobile work productivity	Using the mobile service in my job reduces travelling from and to the office during the work day I can complete my work tasks in less time than before due to the mobile service. Using the mobile service in my job increases my productivity.

4.3.1.3 Questions to Determine Viability and Fit

A final series of statements were developed to be included on both the ShedVIEW and the Data Collection Application user surveys. Because the topic of viability will be primarily addressed through a separate developer-survey, the majority of the statements presented in this

section regard the topic of fit. Fit is best addressed given five attributes that indicate a solution's fitness as follows:

- *Ubiquity*: Available at any location at any time;
- *Convenience*: Convenient for users to operate;
- *Instant connectivity*: Easily connected to the target;
- *Personalization*: Allows for preparation of personalized information;
- *Localization*: Location-specific information and products (19)."

Question statements, presented in Table 4.3, were developed within the applicable categories of ubiquity, convenience, and instant connectivity. Personalization was excluded from the questionnaire because neither ShedVIEW nor the Data Collection Application allow for the personalization of the application. Localization was also excluded as its own category from the questionnaire, but was addressed in the topic of Instant Connectivity.

The topic of viability was limited to two questions under the topic of likeability. These questions are meant to obtain information regarding the extent to which users enjoy using the applications and if they would use it given a choice. This information obtained from these answers will be used in the developer-survey as discussed in later sections.

TABLE 4.3 Statements to Determine Viability and Fit

Indicator	Theme	Statements
Fit	Ubiquity	I can use the application regardless of my location
	Convenience	Using the mobile application is convenient Using the mobile application is more convenient than other methods
	Instant Connectivity	Certain functions of the application take too much time I find myself waiting longer than necessary for the application to find my GPS location The application closes unexpectedly
Viability	Likability	I find this application useful I enjoy using the mobile device as part of my work activities

4.3.2 Final Survey

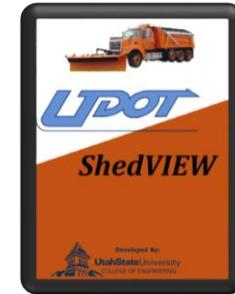
The statements in the tables above were combined to create two surveys, one for each application. The ShedVIEW final survey, presented in Figure 4.2, is a combination of twenty-one different questions from within each of the four categories previously discussed. The slightly longer final survey for the Data Collection Application is presented in Figure 4.3 combines twenty-five different questions from the same categories. Both surveys can be administered using either paper or electronic methods depending on the technology available to the mobile users. While neither survey contains highly sensitive questions, care should be taken in order to maintain anonymity throughout the completion and submission process. This will help to ensure results are not biased by any fear of repercussion, regardless of whether or not the fear is substantiated.

FIGURE 4.2 ShedVIEW Final Survey

Please answer the questions below:

Which age group represents you?	<input type="checkbox"/> 21 and Under	<input type="checkbox"/> 22-34	<input type="checkbox"/> 35-44	1
	<input type="checkbox"/> 45-64	<input type="checkbox"/> 65+		
What is your highest level of education?	<input type="checkbox"/> None	<input type="checkbox"/> High School or GED	<input type="checkbox"/> Some College	2
	<input type="checkbox"/> Associates Degree	<input type="checkbox"/> Bachelors Degree	<input type="checkbox"/> Graduate Degree	
What best describes your employment position with UDOT?	<input type="checkbox"/> Field Personnel	<input type="checkbox"/> Engineering	<input type="checkbox"/> Manager	3

Employee Survey:
ShedVIEW Mobile



Select the best answer to the statements below:

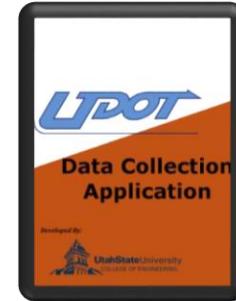
	Strongly Disagree	Disagree	Undecided	Agree	Strongly Agree	
The mobile service was easy to install onto the mobile device	<input type="checkbox"/>	1				
Early experience with other mobile services (or products) made it easier to operate with this service	<input type="checkbox"/>	2				
The organization of information in the mobile service is clear	<input type="checkbox"/>	3				
It is easy to navigate between hierarchical menus, ages, and inside of each screen with the mobile service	<input type="checkbox"/>	4				
The response times are fast enough in the mobile service	<input type="checkbox"/>	5				
The mobile service enables quick, effective, and economical performance of work tasks	<input type="checkbox"/>	6				
I would recommend the mobile service also for others doing the same work	<input type="checkbox"/>	7				
I always know where to look for help if I have problems in performing work tasks with the mobile service or device	<input type="checkbox"/>	8				
Using the mobile service in my job reduces travelling from and to the office during the workday	<input type="checkbox"/>	9				
Using the mobile service in my job increases my productivity	<input type="checkbox"/>	10				
I can use the application regardless of my location	<input type="checkbox"/>	11				
Using the mobile application is convenient	<input type="checkbox"/>	12				
Using the mobile application is more convenient than other methods	<input type="checkbox"/>	13				
Certain functions of the application take too much time	<input type="checkbox"/>	14				
I find myself waiting longer than necessary for the application to find my GPS location	<input type="checkbox"/>	15				
The application closes unexpectedly	<input type="checkbox"/>	16				
I find this application useful	<input type="checkbox"/>	17				
I enjoy using the mobile device as part of my work activities	<input type="checkbox"/>	18				
	Strongly Disagree	Disagree	Undecided	Agree	Strongly Agree	

FIGURE 4.3 Data Collection App Final Survey

Please answer the questions below:

Which age group represents you?	<input type="checkbox"/> 21 and Under	<input type="checkbox"/> 22-34	<input type="checkbox"/> 35-44	1
	<input type="checkbox"/> 45-64	<input type="checkbox"/> 65+		
What is your highest level of education?	<input type="checkbox"/> None	<input type="checkbox"/> High School or GED	<input type="checkbox"/> Some College	2
	<input type="checkbox"/> Associates Degree	<input type="checkbox"/> Bachelors Degree	<input type="checkbox"/> Graduate Degree	
What best describes your employment position with UDOT?	<input type="checkbox"/> Field Personnel	<input type="checkbox"/> Engineering	<input type="checkbox"/> Manager	3

Employee Survey:
Data Collection Application



Select the best answer to the statements below:

	Strongly Disagree	Disagree	Undecided	Agree	Strongly Agree	
I can complete my work tasks quickly by using the service with a mobile device	<input type="checkbox"/>	1				
Some work tasks are too slow to complete with the mobile service	<input type="checkbox"/>	2				
The work task sometimes fails because of the mobile service	<input type="checkbox"/>	3				
The mobile service enables quick, effective and economical performance of work tasks	<input type="checkbox"/>	4				
I would recommend the mobile service also for others doing the same work	<input type="checkbox"/>	5				
The battery capacity of the mobile device is sufficient for the use of the service for work tasks	<input type="checkbox"/>	6				
The screen size of the mobile device is adequate for using the service for work tasks	<input type="checkbox"/>	7				
Inputting information to the mobile device is easy	<input type="checkbox"/>	8				
Sometimes, the environment (such as coldness, sunshine, darkness) makes the use of the service difficult	<input type="checkbox"/>	9				
I always know who to ask for help if I have problems in performing work tasks with the mobile service or device	<input type="checkbox"/>	10				
Using the mobile service in my job reduces travelling from and to the office during the work day	<input type="checkbox"/>	11				
I can complete my work tasks in less time than before due to the mobile service	<input type="checkbox"/>	12				
Using the mobile service in my job increases my productivity	<input type="checkbox"/>	13				
Using the mobile service in my job increases my work motivation	<input type="checkbox"/>	14				
I can use the application regardless of my location	<input type="checkbox"/>	15				
Using the mobile application is convenient	<input type="checkbox"/>	16				
Using the mobile application is more convenient than other methods	<input type="checkbox"/>	17				
Certain functions of the application take too much time	<input type="checkbox"/>	18				
I find myself waiting longer than necessary for the application to find my GPS location	<input type="checkbox"/>	19				
The application closes unexpectedly	<input type="checkbox"/>	20				
I find this application useful	<input type="checkbox"/>	21				
I enjoy using the mobile device as part of my work activities	<input type="checkbox"/>	22				
	Strongly Disagree	Disagree	Undecided	Agree	Strongly Agree	

4.3.3 Determining Viability

A final measure of technology effect is given in the topic of viability. Viability, as discussed in the literature review portion of this report, is best measured through a series of introspective developer-based questions not suitable for inclusion in the surveys above. Honest answers to these questions by application developers or researchers will indicate whether an application is able to maintain viability. The questions are grouped into three categories: financial costs, transaction costs, and accommodating factors.

Financial costs refer to the monetary impact of an application within an organization. Four statements that can be used to determine this impact are as follows:

1. The target application is a frequent business process
2. The transaction process involves high uncertainty
3. The current equipment and technological architecture is adequate for supporting the application
4. The cost for using the wireless service is high (18)

These statements are to be assessed using a Likert scale with the intent of the scale results being to provide information on the application's degree of conformity to each of the questions. Ideally, the application should be used in a frequent business process, not involve a high uncertainty, have support from pre-existing technology architecture, and have a low-cost.

An additional indicator of viability involves the user-transaction cost that comes are a result of the application. Assessed similarly to the financial costs, transaction costs are determined from the following statements:

1. A great deal of time and effort are necessary for the user to learn the application
2. The application requires a reallocation of resources (18)

The ideal application would require little time to learn and minimal reallocation of business resources.

A final measure of viability is used to measure the applications acceptance with the idea that a well-accepted application may still be considered viable despite having a

higher cost. This acceptance is determined through four questions assessed in the same manner as both of the cost indicators. The statements are as follows:

1. Top management supports the application
2. The popularity of the intended mobile devices is high
3. People like to use the intended mobile device
4. There is a popular substitute service available (18)

Ideal answers to the statements above would indicate management support for the application, a popularity of the mobile devices, user-acceptance, and no popular substitutes. In addressing questions three and four above, respondents should refer to the application-specific survey questions regarding viability as discussed previously.

Each of the statements above should be assessed after the user-surveys have been circulated using the survey in Figure 4.4 but prior to the data analysis. In responding to this survey, developers need to keep an open mind and strive to remain unbiased in their assessment.

FIGURE 4.4 Viability Survey

Mobile Application: _____

Respondent: _____

Select the best answer to the statements below:

	Strongly Disagree	Disagree	Undecided	Agree	Strongly Agree	
The target application is a frequent business process	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1
The transaction process involves high uncertainty	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	2
The current equipment and technological architecture is adequate for supporting the application	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	3
The cost for using the wireless service is high	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	4
A great deal of time and effort are necessary for the user to learn the application	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	5
The application requires a reallocation of resources	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	6
Top management supports the application	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	7
The popularity of the intended mobile devices is high	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	8
People like to use the intended mobile device	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	9
There is a popular substitute service available	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	10
	Strongly Disagree	Disagree	Undecided	Agree	Strongly Agree	

4.3.4 Survey Administration Timetable

Survey administration can occur at a number of different stages after application implementation. This primarily depends upon UDOT's acceptance and willingness to support and endorse survey efforts. Ideally, the survey will be administered after application users are able to adjust to the new system. This will ensure that the survey results do not reflect upon first impressions, many of which turn out to be unsubstantiated. Additionally, if analysis beyond that which is presented in this report is desired, two survey iterations could be implemented. The first survey would be administered directly after the application deployment and the second given once users are able to adjust to the new technology. Analysis of data from both surveys would highlight any initial misconceptions held by the user population regarding mobile technology.

4.4 Survey Result Analysis

Once the survey results are obtained, an analysis needs to be performed in order to determine the either positive or negative effect of the applications. This analysis is carefully formulated using a combination of descriptive statistics and statistical inference to describe various phenomenon occurring within the survey result data. The sections below present a method of analysis that can be used given the surveys presented above.

4.4.1 Data Preparation

Once all of the survey responses have been submitted, or the time to return the surveys has expired, four instances of data will have been obtained. Two instances are obtained from the ShedVIEW and the Data Collection Application surveys. The other two instances are obtained from the developer-based viability survey that is assessed for both applications. Each instance of survey data needs to be analyzed separately. Because the viability survey only contains one response for each of the surveys, it will be analyzed separate from the application surveys. All data instances, however, will need to be cleaned, corrected, and defined using processes that are described in the following paragraphs.

4.4.1.1 Cleaning Data Set

The data obtained from the surveys presented above is straightforward and requires little cleaning. The data should be reviewed, however, to ensure that each of the ratings lies between the numbers one and five, the max and min of the rating scale. Additionally, the data should be reviewed for obviously falsified data such as an entire survey returning a single number response. An aid in identifying this phenomenon is the final questions from each survey whose scales are inverted from the other questions. In general, these questions responses should be lower for higher responses to previous questions and vice-versa.

An additional cleaning step required for analysis is the conversion of the survey responses to a numerical scale. This conversion, shown in Tables 4.4 and 4.5, can be either initially defined in a software-based survey, or manually converted at the time of data entry. Either case requires care to ensure correct values are assigned to each of the original responses. Additionally, in order to minimize the chance of errors in later analysis, the demographic responses are converted to numbers outside of the range of the likert-scale converted responses.

TABLE 4.4 Response Conversion Values (Likert)

Original Response	Conversion
Strongly Disagree	1
Disagree	2
Undecided	3
Agree	4
Strongly Agree	5

TABLE 4.5 Response Conversion Values (Demographic)

Question	Original Response	Conversion
Which age group represents you?	21 and Under	11
	22-34	22
	35-44	33
	45-54	44
	65+	55
What is your highest level of education?	None	11
	High School or GED	22
	Some College	33
	Associates Degree	44
	Bachelors Degree	55
	Graduate Degree	66
What best describes your employment position with UDOT?	Field Personnel	11
	Engineering	22
	Manager	33

4.4.1.2 Correcting Data

Some of the questions from both surveys require the responder to answer on a scale that has an opposite meaning than it does for other questions. These questions, presented in Table 4.6, require a correction to allow an analysis to be performed. For each of the questions requiring this correction, responses should be modified according to the values presented in Table 4.7. This will ensure that each of the questions has the same scale and as such, result in data appropriate for statistical analysis.

TABLE 4.6 Questions Requiring Correction

	Question Number
ShedVIEW	14, 15, 16
Data Collection	2, 3, 9, 18, 19, 20
Viability	2, 4, 5, 6, 10

TABLE 4.7 Response Correction Values

Original Response	Correction
1	5
2	4
3	3
4	2
5	1

4.4.1.3 Defining Data Categories

The last preparation step is to combine the survey questions into groups that describe the evaluation categories of usability, productivity, fit, and viability. This is done using Tables 4.1, 4.2, and 4.3 above. While simple, this step will ensure that the data can be defined and grouped as needed in analysis efforts.

4.4.2 Data Analysis Using Likert Scales

Data analysis performed on data obtained from Likert scales presents a controversial issue due to the typically ordinal nature of the data. The debate goes both ways depending upon the research being studied. According to one article, a possible reason for this contention may lie in the fact that treating the data as non-ordinal opens the door to a variety of powerful statistical applications (32). If the data is to be considered ordinal, any combination of parametric statistical analysis is not appropriate. If it is not considered ordinal, the analysis results can be both misleading and misrepresenting (32). Because the data obtained from the application

surveys is truly ordinal in nature, the analysis framework presented in the following sections will involve only non-parametric statistical methods.

4.4.3 Analysis Framework

A statistical analysis shall be performed on both instances of the user-based survey data in order to find the applications effect using the four indicators mentioned above as determining factors. This analysis is performed using descriptive statistics which are “procedures for organizing and summarizing data so that the important characteristics can be described and communicated (46).” As mentioned above, this descriptive analysis will be somewhat different than typical parametric-based analysis given the data obtained from the likert scales is ordinal in nature. Each component of the analysis is presented in the sections below. Analysis for the developer-based viability survey data will be presented in later sections.

To provide a better description of the procedures that should be taken in the analysis of survey data, the following paragraphs also contain the results of calculations discussed below performed on mock data. This data, shown in its corrected form in Table 4.8, was obtained from users who were told to think of an application and try their best to respond and as such has no bearing. It is, however, useful as an example and in the explanation of various phenomenon that may occur once the actual surveys are implemented.

TABLE 4.8 Sample Data for ShedVIEW Survey

	Question #	Question	Corrected Responses							
			1	2	3	4	5	6	7	8
Usability	1	Which age group represents you	22	22	22	22	22	22	33	22
	2	What is your highest level of education	66	66	55	66	66	55	44	66
	3	What best describes your employment position with UDOT	22	22	22	22	11	11	11	11
	1	The mobile service was easy to install onto the mobile device	5	4	4	4	5	4	3	5
	2	Early experience with other mobile services (or products) made it easier to operate with this service	5	4	4	4	4	2	4	5
Productivity	3	The organization of information in the mobile service is clear	3	3	5	3	4	3	3	5
	4	It is easy to navigate between hierarchical menus, ages, and inside of each screen with the mobile service	4	4	4	4	5	4	5	5
	5	The response times are fast enough in the mobile service	3	3	3	4	3	5	5	4
	6	The mobile service enables quick, effective, and economical performance of work tasks	2	3	5	5	5	4	5	2
	7	I would recommend the mobile service also for others doing the same work	2	3	4	5	5	4	5	5
Fit	8	I always know where to look for help if I have problems in performing work tasks with the mobile service or device	2	4	3	4	3	2	2	2
	9	Using the mobile service in my job reduces travelling from and to the office during the workday	3	4	4	4	4	4	4	2
	10	Using the mobile service in my job increases my productivity	2	3	4	5	5	4	3	5
Viability	11	I can use the application regardless of my location	5	4	3	2	1	3	3	1
	12	Using the mobile application is convenient	3	2	2	2	1	2	3	1
	13	Using the mobile application is more convenient than other methods	3	3	2	3	2	2	2	2
	14	Certain functions of the application take too much time	3	3	2	4	3	2	4	4
	15	I find myself waiting longer than necessary for the application to find my GPS location	3	3	2	4	4	4	2	1
	16	The application closes unexpectedly	3	4	2	2	2	3	2	2
Viability	17	I find this application useful	3	4	5	4	4	5	4	5
	18	I enjoy using the mobile device as part of my work activities	3	3	4	5	4	5	4	4

4.4.3.1 Frequency Distribution

Individual scores such as those obtained from Likert data are best organized using a frequency distribution that indicates the number of times a given score or rating occurs within a data set. This distribution provides the answers to two questions at once: which ratings were used and how often were they used (46). Once complete, the distribution can be used to both visually and quantitatively to represent various response trends. Organizing the data by frequency is one of the most effective ways of determining descriptive trends within the survey data.

Frequency distributions need to be created for each of the survey categories as well as all of the data combined. This can be done in spreadsheet software such as Microsoft Excel using the COUNTIF function. Given each of the Likert ratings one through five, simply count the number of occurrences within each category of questions relating to usability, productivity, fit, viability, and finally extend the count to all of the categories combined. To facilitate clarity, a relative frequency distribution can then be created by dividing the number of categorical occurrences by the total occurrences. This results in a percentage of the total scores for each response.

The results of this frequency analysis performed on the sample data are shown in Figure 4.5 below in terms of relative frequency. It can be seen from the figure that overall, the responses are skewed to the left indicating that the application is effective (disregarding viability which will be analyzed separately). It can also be seen that the area of fit is skewed to the right indicating that either these responses were not converted correctly or the application did not have a good fit within the organization.

To refine the results the data can be separated by age, education level, and employment position. Once grouped, a frequency analysis as described above can then be performed within each of the demographic categories. This type of analysis will help to identify group-based trends and phenomenon that is not accounted for in higher-level analysis. When breaking the data down into groups, care must be taken to retain a high enough sample size that the data can still be considered representative. The determination of this minimum representative sample size will be left to engineering judgment (47).

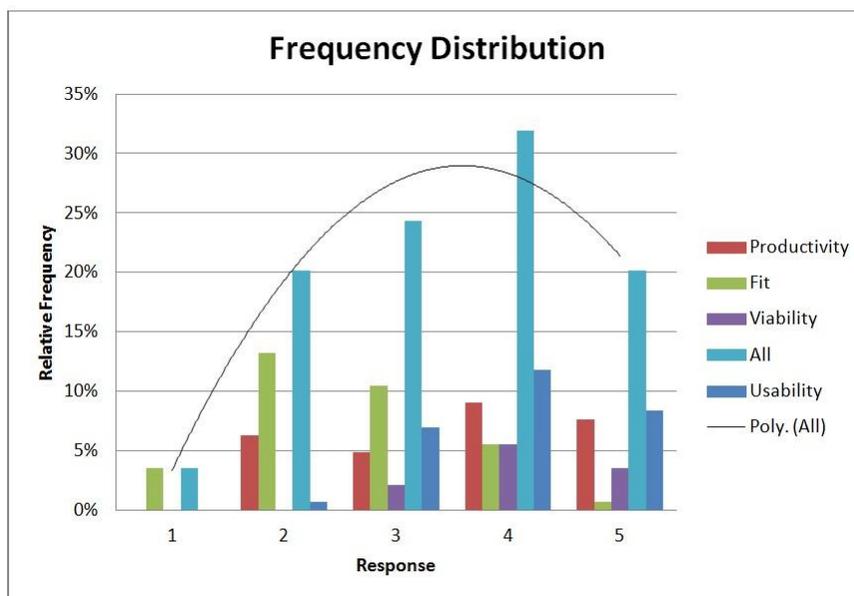
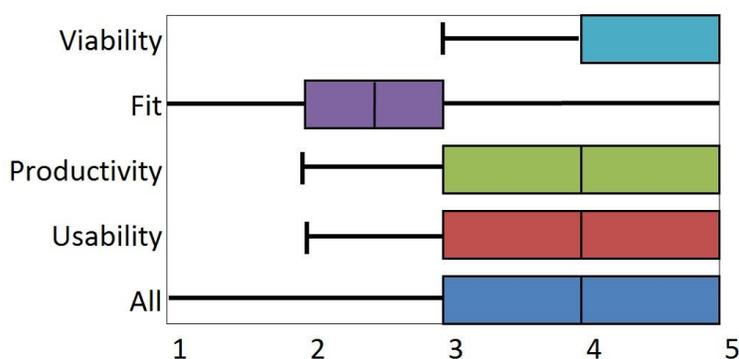


FIGURE 4.5 Frequency Distribution for Sample Data *Additional Descriptive Statistics*

To further describe various trends existing within the survey-derived data, additional statistical variables can be used. The mode is the best descriptor of the central tendency of ordinal data and represents the most frequently occurring score (46). Additional descriptive statistics that can be used include the median, min, max, and quartiles one and three. The median presents the data value at the 50th percentile, and the quartiles represent the 25th and 75th percentile respectively. Each of these values can be used to describe trends in central tendency and are best implemented using a box and whisker plot as shown in Figure 4.6. Given the data presented in the example box and whisker plot, it can be seen that viability is the most compliant measure of effectiveness and fit needs to be addressed. As with the frequency distribution analysis, this analysis can also be extended to demographic groups to identify group-specific trends.

TABLE 4.9 Sample Data Statistics

	Mode	Min	Quartile 1	Median	Quartile 3	Max
All	4	1	3	4	4	5
Usability	4	2	3	4	5	5
Productivity	4	2	3	4	5	5
Fit	2	1	2	2.5	3	5
Viability	4	3	4	4	5	5

**FIGURE 4.6 Box and Whisker Plot for Sample Data Viability Survey Analysis**

Analysis is performed on the developer-based viability survey data using the same techniques as presented for the user-survey data. The difference is that this is a single instance of data and as such, the data needs to be analyzed separately to avoid misinterpretation. As with the user-survey data, a survey was administered to provide an example for clarification. Because this survey in general only requires the applications to be completed, this survey represents actual data for the ShedVIEW application. This survey, presented in Table 4.1, shows a single respondents answers as well as the correction. This data is then analyzed through a frequency distribution in Figure 4.7 and a box and whisker plot in Figure 4.8. Conclusions that can be drawn from this data are presented in the following section.

TABLE 4.10 Responses to the ShedVIEW Viability Survey

Question #	Question	Original Response	Corrected Response
1	The target application is a frequent business process	5	5
2	The transaction process involves high uncertainty	1	5
3	The current equipment and technological architecture is adequate for supporting the application	2	2
4	The cost for using the wireless service is high	1	5
5	A great deal of time and effort are necessary for the user to learn the application	1	5
6	The application requires a reallocation of resources	2	4
7	Top management supports the application	5	5
8	The popularity of the intended mobile devices is high	4	4
9	People like to use the intended mobile device	4	4
10	There is a popular substitute service available	1	5

TABLE 4.11 Viability Data Statistics

	Mode	Min	Quartile 1	Median	Quartile 3	Max
Viability	5	2	4	5	5	5

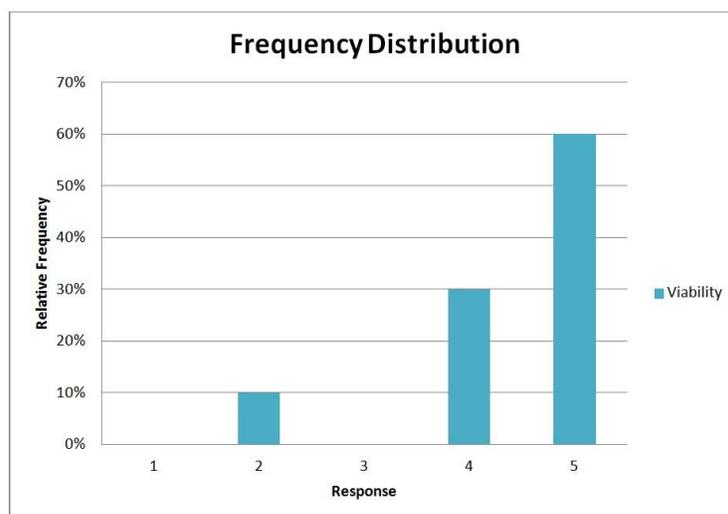


FIGURE 4.7 Frequency Distribution for Viability Data

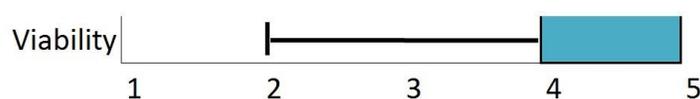


FIGURE 4.8 Box and Whisker Plot for Sample Data *Interpreting Results*

Determining the applications effect needs to first be viewed on a broad overall scale using the frequency distributions and then refined through the use of central tendency indicators including the box and whisker plots. Further refinement and group comparison occurs using the inference method discussed in the following section. In general, an analysis indicating success or mobile effectiveness should reflect the greater majority of scores within each category as greater than three. In the sample analysis results above, it can be seen from Figure 4.5 that there is an overall trend in the ‘all’ curve to indicate effectiveness, but the individual curves reveal problems in the areas of fit and productivity. The mode, an indicator that shows central tendency, presented in Table 4.9 verifies this interpretation. Finally, from the box and whisker plots Figure 4.6 and 4.8 as well as the frequency plot in Figure 4.7, it can be seen that the application excelled in viability. Alternately, Figure 4.6 shows that fit is the worst performing indicator and also acts to clarify the indicator of productivity as within the acceptable region. In conclusion, it can be determined from this sample analysis that the application performed on a whole fairly well, but cannot be deemed effective because it failed in the area of fit. As it is, the ‘sample’ application needs to be reworked to address the performance indicator of fit.

4.4.4 Statistical Inference Framework

While the descriptive statistics calculated and displayed above do a good job of presenting the data, further statistical analysis is needed for comparisons regarding demographic groups. This analysis is performed using a non-parametric statistical process described below.

4.4.4.1 Goodness of Fit Test

The Chi Square test is referred to as a goodness of fit test because it measures the relationship between different categories. This test addresses the following question: “As the categories change, do the frequencies with which participants fall in the categories also change (46)?” In the case of the survey data obtained, the Chi Square test will help to determine if the responses among varying demographic groups are different. To perform this test, a null hypothesis needs to be established. The typical null hypothesis is reflected below:

$$H_0: \text{All frequencies in the population are equal}$$

Similarly, an alternate hypothesis needs to be formed that implies that a direct relationship was not demonstrated. This alternate hypothesis is presented below:

$$H_1: \text{Not all frequencies in the population are equal}$$

Once the null and alternate hypotheses have been established, the data can be arranged in a way that analysis can be performed. The Chi Square test can be used to compare answers in a number of different ways depending on the amount of data that is available for analysis. The first comparison is made between all survey responses within two demographic groups. This can be further separated to identify trends within the four effectiveness indicator categories. One might, for example, compare responses within the indicator of Viability between the employment groups of Engineer and Manager. The final arrangement of the sample data presented in Table 4.12 below displays the first comparison mentioned for the employment categories of Field Personnel and Engineer. This arrangement simply calculates the response frequency within each of the four indicators (entire survey) and then adds them together by row and column to produce totals. Once the calculated frequency values have been arranged, the expected values for each category need to be calculated. This is done using the formula below:

$$Exp_{i,j} = \frac{\text{sum}(i:i)}{\text{sum}(i:j)} * \text{sum}(j:j)$$

Where $i = \text{Response Category}$ and $j = \text{Demographic Category}$

The expected values need to be inspected to ensure that the results of the Chi Square test will be valid. Generally, no single cell should have a frequency less than one, and no more than 20% of the expected values should be less than five (46).

TABLE 4.12 Chi-Square Calculated Values

Response	1	2	3	4	5	Totals
Field Pers.	5	16	11	21	19	72
Engineer	0	13	24	25	10	72
Totals	5	29	35	46	29	144

TABLE 4.13 Chi-Square Expected Values

Response	1	2	3	4	5	Totals
Field Pers.	2.5	14.5	17.5	23	14.5	72
Engineer	2.5	14.5	17.5	23	14.5	72
						144

Upon completion and inspection of the expected value table, the function CHITEST can be used in Excel to perform the calculations required. This function produces a p value which is compared against the value of .05 to either reject or accept the null hypothesis. Given an alpha value of .05, then a value of $p < .05$ results in the null hypothesis being rejected (46). In this case, additional hypotheses can be tested against the previously alternate hypothesis to refine the statistical inference.

In performing the sample calculations presented in the tables above, a value of 0.009986 was returned from the Chi Square analysis. This value indicates that a discrepancy occurs between the two employment categories analyzed and an investigation into why such a discrepancy occurs needs to occur.

4.4.5 Sample Size

The sample size needed for the calculations outlined above is assumed to be small. This assumption is acceptable provided that the following conditions are met:

1. The sample must be representative meaning that judgment must be used to ensure that the group of employees surveyed effectively represents the whole population.
2. The robustness of statistical tests requires more than five samples (minimum) per group.
3. The sample size needs to be large enough to maintain statistical significance (47).

If these conditions are not met a larger group needs to be surveyed, or survey techniques and response enticements need to be modified. It is anticipated that if the surveys receive support from upper level management within UDOT, soliciting responses will not be problematic.

4.5 Feedback Loop

Upon completion of the analysis above, the actual effects of mobile technology within UDOT will have been determined. Using these results, the applications need to be updated to reflect any areas of potential concern. If there are substantial problem areas that clearly affected the outcome of the survey, additional survey iterations may need to be completed after the applications have been corrected and updated. This iterative framework is best displayed in Figure 4.9. Any future survey activities should use prior surveys as a baseline to ensure improvement is being made and the outcome of the improvements is as expected. The end goal of this feedback process is to provide UDOT with an application that provides an effective mobile solution for their workforce.

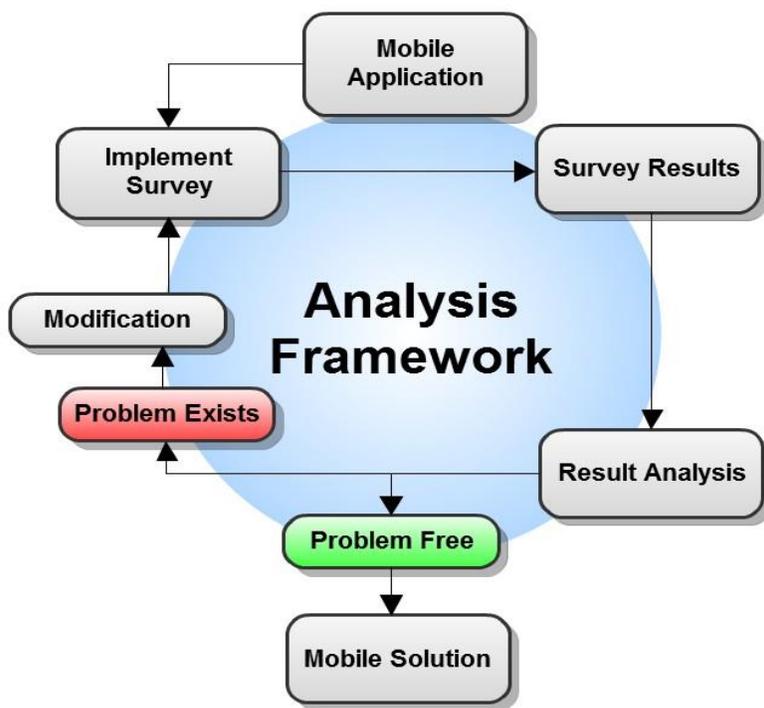


FIGURE 4.9 Feedback Loop Conclusion

The purpose of evaluation is to analyze the usability, viability, fit, and effect on productivity that a mobile application has within an agency. To accomplish this, two applications were developed for UDOT employees to use. These applications, once implemented, will be analyzed using a set of surveys that collects feedback on the four parameters mentioned above. This information will be analyzed in an effort to determine the applications effect within UDOT. If the applications can be determined to be viable, have a good fit, be usable, and contribute to productivity then the implementation has had a positive effect on the agency. If the applications should be lacking in one or more of the aforementioned areas, however, then changes need to occur within the applications in regards to the problem areas in order to reach compliance within each of the indicative areas. Following this process will ensure that UDOT obtains effective mobile solutions that positively influence the agency.

CHAPTER 5

CONCLUSION AND FUTURE EFFORTS

Budget cuts and increased public accountability, along with many other factors have led governmental transportation agencies to seek out alternative methods of addressing their ever-increasing workload. With recent advances in the realm of consumer-based mobile technology, relatively cheap and easy to use devices and software are available to potentially address the need to accomplish more work given less resources. In this report, two mobile applications were developed to address specific areas of concern with the Utah Department of Transportation (UDOT). A framework for analysis was then developed to identify the applications effect as either positive or negative within the agency.

5.1 Research Question Readdressed

The intent of this research is to identify the implementation effect of two customized mobile applications developed for the specific use within the maintenance department of UDOT. The specific question that this research addresses is: given distinct challenges relating to the transfer of information maintained on centralized databases within a DOT, can mobile technology be implemented as a solution? To address this question, application ideas were sought from UDOT management. Their ideas were coupled with an extensive literature analysis in order to provide an appropriate long-term solution. In the end, two mobile applications were developed, namely ShedVIEW and a Data Collection Application. While different in purpose, these applications both specifically address the area of information transfer and retrieval.

As development of the mobile applications progressed, another literature review was performed in the areas of mobile development and process evaluation. Topics from these areas were used to develop surveys, both user and developer based for each application. These surveys are distributed after the applications have been implemented and seek to determine the applications usability, viability, fit, and effect on user productivity. The data collected from these surveys will then be analyzed in order to determine if the applications impact on the agency was either positive or negative. This process from the identification of concern areas to the development of an analysis framework is depicted in Figure 5.1 to provide further clarity.

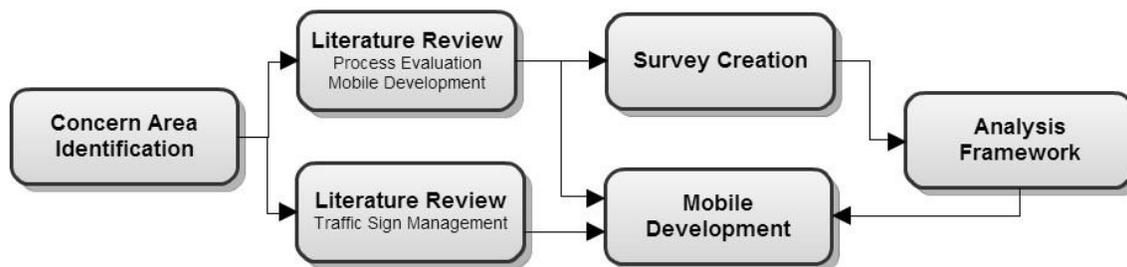


FIGURE 5.1 Research Approach

5.2 Potential Conclusions from Analysis Framework

Once the analysis framework has been implemented, the application can be considered as either positively or negatively impacting the Department of Transportation. A positive impact indicates that the application was effective and a negative impact indicated non-effectiveness. This conclusion is reached given the indicators of usability, viability, fit, and productivity as mentioned above. If the conclusions from the analysis indicate the application is viable, has a good fit, is usable, and contributes to productivity, then it can be determined that the application will have a positive effect on the department. If it is lacking in one or more of these areas, however, then changes need to be made within the applications in order to improve them and obtain compliance within each of the indicator areas. This compliance is reached through an iterative process of correction and re-analysis as presented in Chapter 4. Additionally, if the application is found to be effective overall, but lacking in a specific demographic group, training and minor application modifications should occur as required in order to improve the application effectiveness within the group.

5.3 Future Efforts

A number of future efforts need to be taken in order to enhance the ShedVIEW and Data Collection Applications. These efforts will lead to the enhancement of the applications and will primarily add background processes to automate information delivery and retrieval. Overall, the application users will be unaware that changes have been made to the application and as such, these changes can be done before or after the analysis described in Chapter 4 is complete. The efforts that need to be undertaken are identified in the sections below. It is important to note that additional efforts not discussed below are required based upon the results of the analysis. If the

application is found to be lacking in one of the indicator areas, work to resolve that area takes priority over the future work efforts below.

5.3.1 ShedVIEW Efforts

At its current stage in development, ShedVIEW is a shell that needs to be manually controlled from the backend. This involves manipulating and submitting the database directly within the application. In future efforts, this process needs to be automated to ensure the seamless transition of data from UDOT servers. Updates to maintenance information on UDOT central servers should be recognized by the application that in turn prompts the user to update the device locally. Additionally, data directly from UDOT servers needs to be run through a process that extracts useful information and formats it in a way that it can be directly imported into the application. This requires coding knowledge using database type tools and processes.

Additionally, location specific features were not included in the initial development efforts of this application due to time constraints. The inclusion of the features discussed in Chapter 2 will be a great benefit to UDOT maintenance personnel. Additional features may also be needed as circumstances change, or as areas that would be aided by additional development are identified. As a result, ShedVIEW is a living application that will need to be tweaked and improved after the initial implementation.

Further functionality that can be extended to the ShedVIEW application is to transform the initial Android development across various device platforms, primarily Apple. This transformation is key to obtaining support from management personnel at UDOT, many of whom solely use Apple mobile device. Android was selected as the initial development platform because it the software was straightforward and freely available. Apple development, on the other hand, is a carefully controlled process that requires additional steps of approval prior to application implementation. The distribution of applications to Apple devices is also somewhat more complex than it is for Android devices. As such, care must be taken in the development process to ensure that changes made do not affect the user experience. If changes are made to the application, a separate analysis is highly suggested.

5.3.2 Data Collection Application Efforts

The Data Collection Application requires a number of future efforts for correct implementation. Once implementation is scheduled, the application needs to be slightly customized for use within UDOT. This customization requires changing the application name, splash screen, and other visual changes. The methods and specific custom parameters can be found in the developer documentation section of the Appendices. Once customized, the backend server needs to be set up on a UDOT-owned or controlled computer. This is a non-complex proved process whose the methods are presented in the Appendices. Once the customization and server set up are complete, the application is ready to be deployed.

After the application is deployed, a number of tasks need to be completed to finalize the application. As with the ShedVIEW application, most of the changes made post-deployment are on the backend and as a result, will not be noticed by the application users. The primary task that needs to be accomplished is the linking of the server database set up to receive data submissions to the UDOT central server. This will be a fairly complex process and will involve a number of contracting agencies. While the link is being created, a number of read/write checks need to be also put into place to ensure that the data entering the system is correct. These checks will ensure that the incoming database is formatted correctly and that no erroneous data is written to the central server. The specific checks are complex and as a result will need to be defined by the server administrator.

5.3.3 Non-Specific Efforts

Perhaps the most important future development is a system that allows users of the applications to anonymously suggest improvements or voice concerns with certain features. This is best done in a similar manner to the rating system employed by Google on its Google Play mobile application market. In an ideal comment and rating forum, application users would be allowed to make their own anonymous comments as well as view and rate other user's comments. This not only identifies specific areas of concern for individuals, but also identifies group concerns as users "like" comments made by other users. This type of commenting system should be applied on a simple web page that can only be accessed by users of the applications.

Access can be limited while keeping comments anonymous through the use of a single password for all users that is changed on a set-schedule to maintain a secure environment.

REFERENCES

- [1] *UDOT Maintenance Facts*.
<http://www.udot.utah.gov/main/uconowner.gf?n=2253002476382780>. Accessed May 16, 2012, .
- [2] Google, Inc. 41°14'25.11" N 111°58'49.57" W. Google Earth.
- [3] Google, Inc. 40°50'05.85" N 111°54'59.67" W. Google Earth.
- [4] Evans, T. L. Development of Assessment Strategies for Sign Retroreflectivity. 2011.
- [5] Cottrell, W. D., S. Bryan, B. R. Chilukuri, V. Kalyani, A. Stevanovic, and J. Wu. Transportation Infrastructure Maintenance Management: Case Study of a Small Urban City. *Journal of Infrastructure Systems*, Vol. 15, No. 2, May 2009, pp. 120–132.
- [6] Federal Highway Administration. *Asset Management Overview*. U.S. Department of Transportation Office of Asset Management, 2007.
- [7] Cambridge Systems, Inc, Parsons Brinckerhoff Quade & Douglas, Inc., Ray Jorgenson Associates, Inc., and Paul D. Thompson. *Transportation Asset Management Guide*. American Association of State Highway and Transportation Officials, Nov. 2002.
- [8] Re, J. M., and P. J. Carlson. Practices to Manage Traffic Sign Retroreflectivity. *NCHRP Synthesis of Highway Practice*, No. 431, 2012.
- [9] Federal Highway Administration. *Asset Management primer*.
<http://www.fhwa.dot.gov/infrastructure/asstmgmt/amprimer.pdf>. Accessed May 8, 2012, .
- [10] Kim, Y. R., J. E. Hummer, M. Gabr, D. Johnston, B. S. Underwood, D. J. Findley, and C. M. Cunningham. Asset Management Inventory and Data Collection. Oct. 2009.
- [11] Findley, D. J., C. M. Cunningham, and J. E. Hummer. Comparison of mobile and manual data collection for roadway components. *Transportation Research Part C: Emerging Technologies*, Vol. 19, No. 3, Jun. 2011, pp. 521–540.
- [12] McGee, H. W., and J. A. Paniati. An Implementation Guide For Minimum Retroreflectivity Requirements For Traffic Signs. Apr. 1998.
- [13] Gebauer, J., and M. J. Shaw. Success Factors and Impacts of Mobile Business Applications: Results from a Mobile e-Procurement Study. *International Journal of Electronic Commerce*, Vol. 8, No. 3, Spring 2004, pp. 19–41.
- [14] Nah, F. F.-H., K. Siau, and H. Sheng. The value of mobile applications. *Communications of the ACM*, Vol. 48, No. 2, Feb. 2005, pp. 85–90.
- [15] Sheng, H., F. F. . Nah, and K. Siau. Strategic implications of mobile technology: A case study using value-focused thinking. *The Journal of Strategic Information Systems*, Vol. 14, No. 3, 2005, pp. 269–290.
- [16] Santhanam, R., and E. Hartono. Issues in linking information technology capability to firm performance. *MIS quarterly*, 2003, pp. 125–153.
- [17] Keeney, R. L. Value-focused thinking: Identifying decision opportunities and creating alternatives. *European Journal of Operational Research*, Vol. 92, No. 3, Aug. 1996, pp. 537–549.
- [18] Liang, T.-P., and C.-P. Wei. Introduction to the Special Issue: Mobile Commerce Applications. *International Journal of Electronic Commerce*, Vol. 8, No. 3, Spring 2004, pp. 7–17.

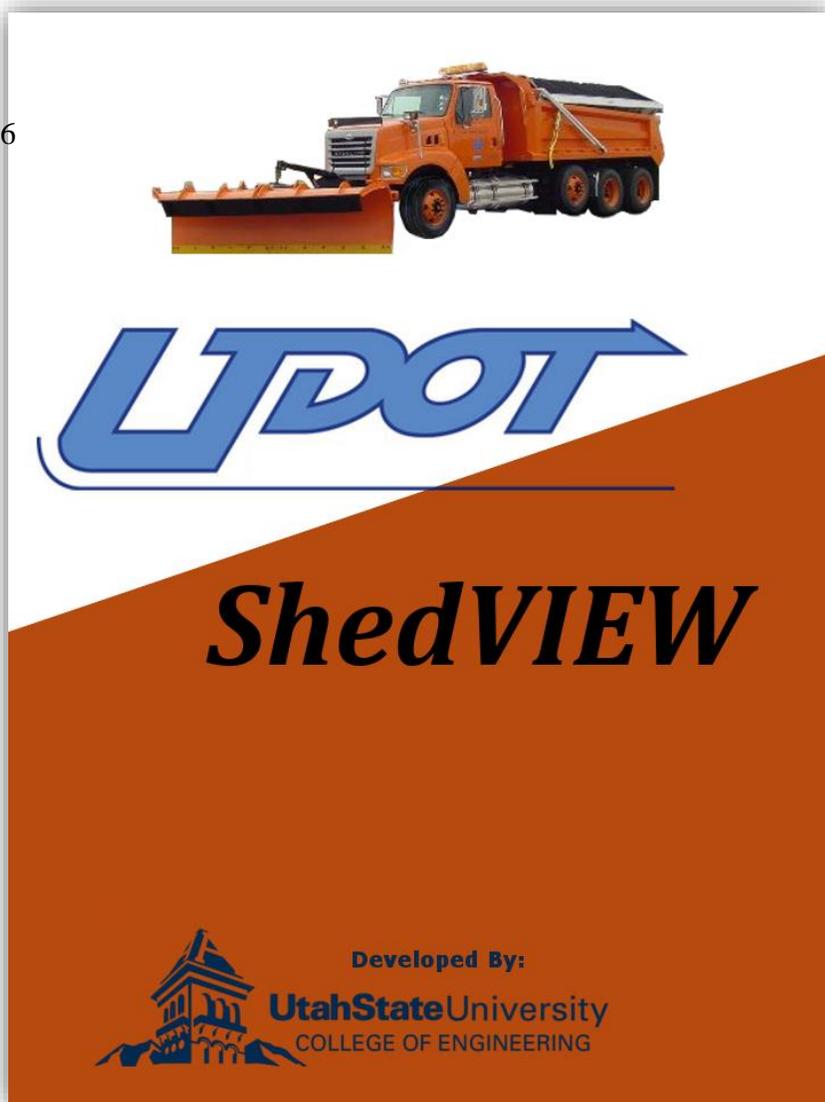
- [19] Turban, E., D. King, and J. Wang. *Introduction to e-commerce*. Prentice Hall Upper Saddle River, New Jersey, 2003.
- [20] Liang, T. P., C. W. Huang, Y. H. Yeh, and B. Lin. Adoption of mobile technology in business: a fit-viability model. *Industrial management & data systems*, Vol. 107, No. 8, 2007, pp. 1154–1169.
- [21] ISO 9241-11. *Ergonomic requirements for office work with visual display terminals - part 11: guidance on usability*. 1998.
- [22] Markova, M., A. Aula, T. Vainio, H. Wigelius, and M. Kulju. MoBiS-Q: a tool for evaluating the success of mobile business services. New York, NY, USA, 2007.
- [23] Willcocks, L. P., and S. Lester. In search of information technology productivity: assessment issues. *Journal of the Operational Research Society*, Vol. 48, No. 11, 1997, pp. 1082–1094.
- [24] Tambe, P., and L. M. Hitt. The productivity of information technology investments: New evidence from IT labor data. *Information Systems Research*, 2012.
- [25] Hannula, M. Total productivity measurement based on partial productivity ratios. *International Journal of Production Economics*, Vol. 78, No. 1, Jul. 2002, pp. 57–67.
- [26] Vuolle, M., M. Tiainen, T. Kallio, T. Vainio, M. Kulju, and H. Wigelius. Developing a questionnaire for measuring mobile business service experience. New York, NY, USA, 2008.
- [27] Zhang, D., and B. Adipat. Challenges, Methodologies, and Issues in the Usability Testing of Mobile Applications. *International Journal of Human-Computer Interaction*, Vol. 18, No. 3, Jul. 2005, pp. 293–308.
- [28] Giner, P., C. Cetina, J. Fons, and V. Pelechano. Implicit interaction design for pervasive workflows. *Personal and Ubiquitous Computing*, Vol. 15, No. 4, 2011, pp. 399–408.
- [29] Gliem, J. A., and R. R. Gliem. Calculating, interpreting, and reporting Cronbach’s alpha reliability coefficient for Likert-type scales. No. 88, 2003.
- [30] McIver, J. P., and E. G. Carmines. *Unidimensional scaling*. Sage Publications, Inc, 1981.
- [31] Mogeey, N. So you want to use a Likert scale. *Learning technology dissemination initiative*, Vol. 25, 1999.
- [32] Allen, I. E., and C. A. Seaman. Likert scales and data analyses. *Quality Progress*, Vol. 40, No. 7, 2007, pp. 64–65.
- [33] Ryan, M. The Likert Scale’s Midpoint In Communications Research. *Journalism Quarterly*, Vol. 57, No. 2, Summer80 1980, pp. 305–313.
- [34] Herzog, A. R., and J. G. Bachman. Effects of questionnaire length on response quality. *Public Opinion Quarterly*, Vol. 45, No. 4, 1981, pp. 549–559.
- [35] Mond, J. M., B. Rodgers, P. J. Hay, C. Owen, and P. J. V. Beumont. Mode of delivery, but not questionnaire length, affected response in an epidemiological study of eating-disordered behavior. *Journal of clinical epidemiology*, Vol. 57, No. 11, 2004, pp. 1167–1171.
- [36] Knapp, H., and S. A. Kirk. Using pencil and paper, Internet and touch-tone phones for self-administered surveys: does methodology matter? *Computers in Human Behavior*, Vol. 19, No. 1, 2003, pp. 117–134.
- [37] Vogel, L. *Android Development Tutorial*.
<http://www.vogella.com/articles/Android/article.html>. Accessed Jun. 18, 2012, .
- [38] Owens, M. *The definitive guide to SQLite*. Apress, 2006.
- [39] Open Data Kit. *About*. <http://opendatakit.org/about/>. Accessed May 2, 2012, .

- [40] Open Data Kit. *Collect*. <http://opendatakit.org/use/collect/>. Accessed Jun. 18, 2012, .
- [41] Open Data Kit. *XLSForm*. <http://opendatakit.org/use/xlsform/>. Accessed Jun. 18, 2012, .
- [42] Formhub. *Syntax*. <http://formhub.org/syntax/>. Accessed Jun. 18, 2012, .
- [43] Open Data Kit. *Aggregate*. <http://opendatakit.org/use/aggregate/>. Accessed Jun. 19, 2012, .
- [44] Open Data Kit. *Deployment Planning*. <http://opendatakit.org/use/aggregate/deployment-planning/>. Accessed Jun. 27, 2012, .
- [45] Tomcat Wiki. *Sites, Applications, and Systems that are Powered By Tomcat*. <http://wiki.apache.org/tomcat/PoweredBy>. Accessed Jun. 27, 2012, .
- [46] Heiman, G. W. *Basic statistics for the behavioral sciences*. Wadsworth Publishing Company, 2010.
- [47] Norman, G. Likert scales, levels of measurement and the “laws” of statistics. *Advances in health sciences education*, Vol. 15, No. 5, 2010, pp. 625–632.
- [48] *Apache License and Distribution FAQ*. <http://www.apache.org/foundation/license-faq.html>. Accessed Aug. 17, 2012, .

APPENDICIES

Appendix A : ShedVIEW User Manual

CHAPTER 6



UDOT ShedVIEW

USER MANUAL

Updated: 9/26/2012

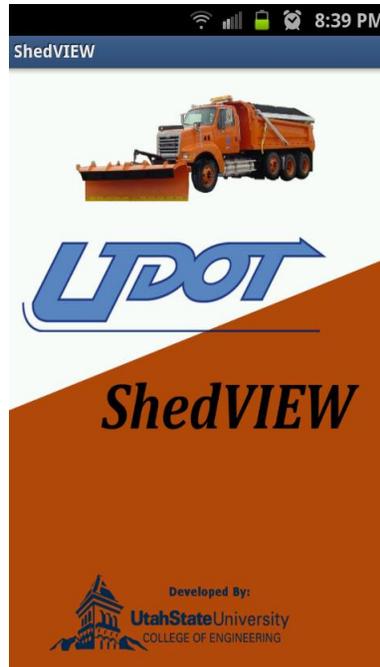
UDOT ShedVIEW

This application provides information about different UDOT maintenance sheds located throughout different regions of Utah. This user manual provides information regarding the usage of application.

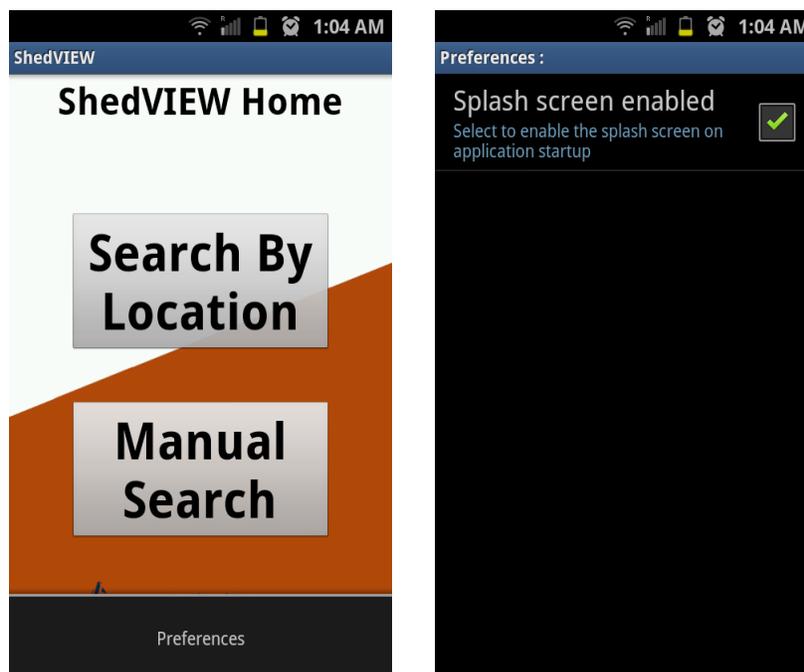
Upon installation of the application, it will appear on the applications screen with the UDOT icon as shown in the figure below.



Splash Screen:

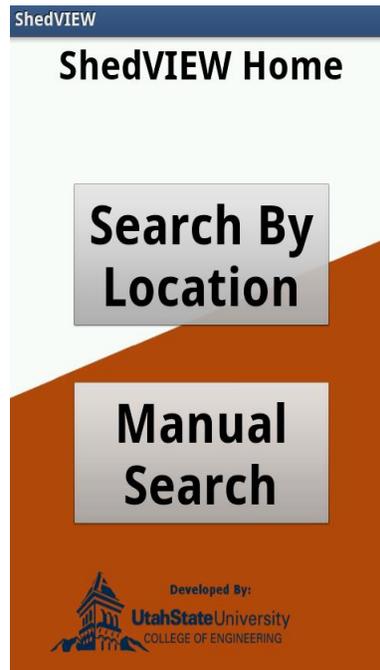


The above splash screen will be displayed at the start of application. If the user does not need splash screen, it can be removed using preferences screen that is accessed from the homepage using the devices menu button as shown in the figures below.



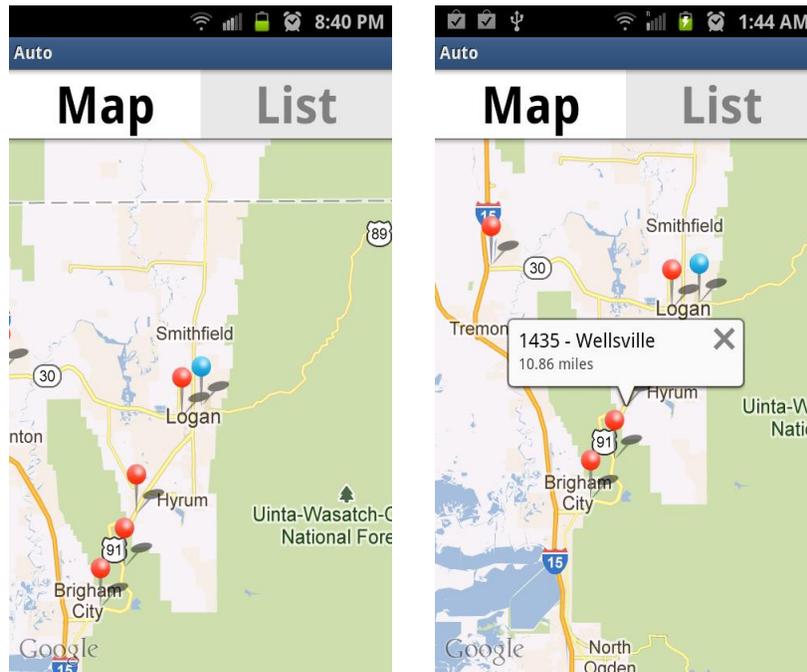
ShedVIEW Home

This is first screen of the application it has two buttons one for manual search and another for location search.



Search by Location (Map)

This screen is a tab view. It consists of two tabs one for map view and one for list view. In the map view screen the application shows the five sheds that are closest to the current user location.



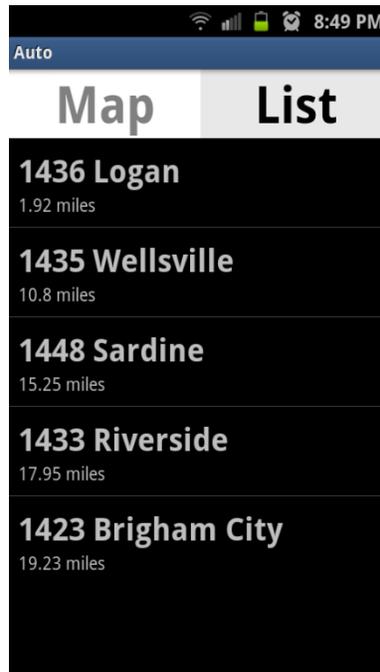
The above screen has map tab selected. The blue color icon is the user current location and red icons show the nearest shed locations.

When the red icon is clicked, the application shows the shed name, city name, and its distance from the current user location as shown in the figure above.

When the overlay view of the red icon is clicked, the shed information screen is displayed.

Search by Location (List)

The list view shows the five nearest shed names in ascending order of distance from the current location. By clicking an item of list view the corresponding shed information screen would be shown.



Manual Search

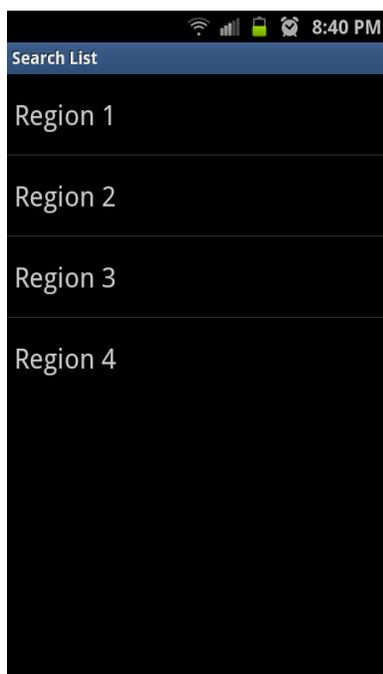
In the manual search screen there are two buttons, one for searching sheds by region and another for searching sheds by city.



Search by Region

This screen shows the different regions available. When any region in the list view is selected, it displays the list of shed in that particular region. This is shown in the figures below.

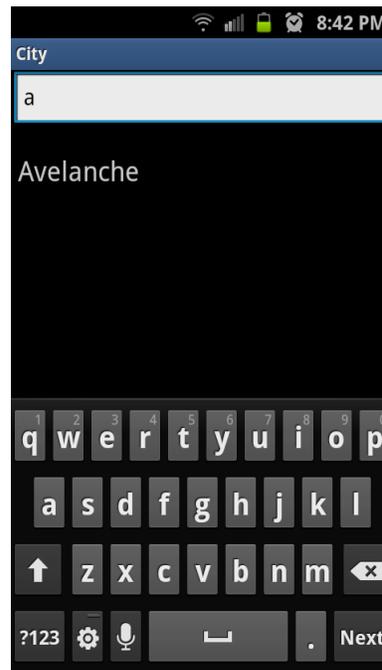
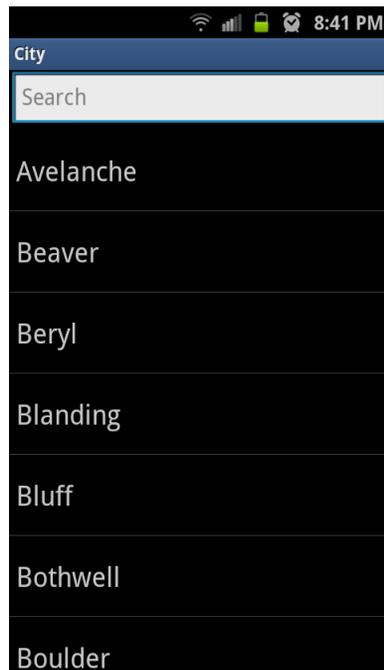
When a station from the list is selected, its corresponding station information screen is shown.



Search by City

This screen shows the list of available cities along with a search box. By using the search box cities can be filtered. Clicking any city in the list view it takes us to sheds available in that particular city. This is shown in the figures below.

By clicking the shed in the list it would take us to the station information screen.

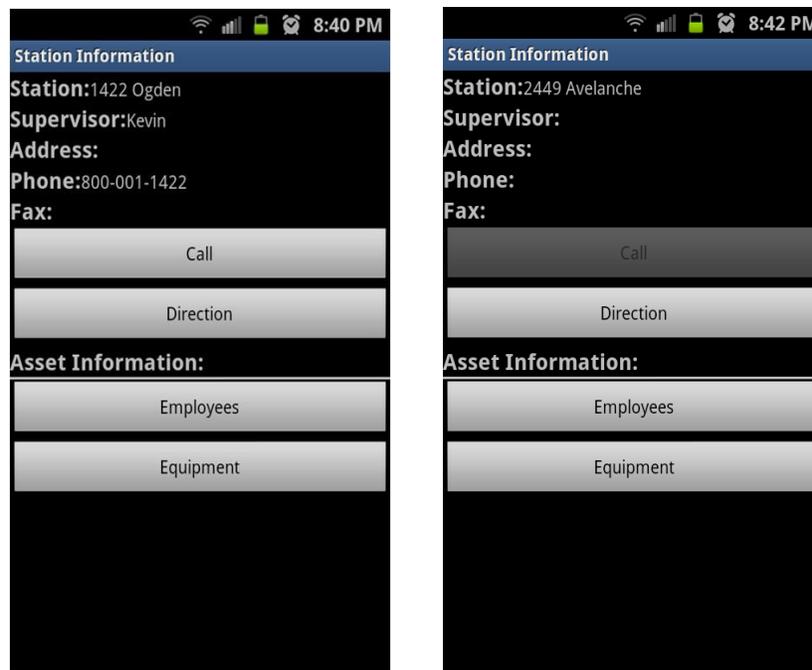


Shed Information:

From various screens in the application, the shed information screen is launched. It contains the shed name, city, address, phone number, and fax number. In addition to above details it also has functions to place a call, obtain directions, and find further information on the Equipment and Employees within that shed.

If any of the above information (call, direction, employees, equipment) is not present in the database then the corresponding button is disabled.

The shed information layout is presented in the figures below.



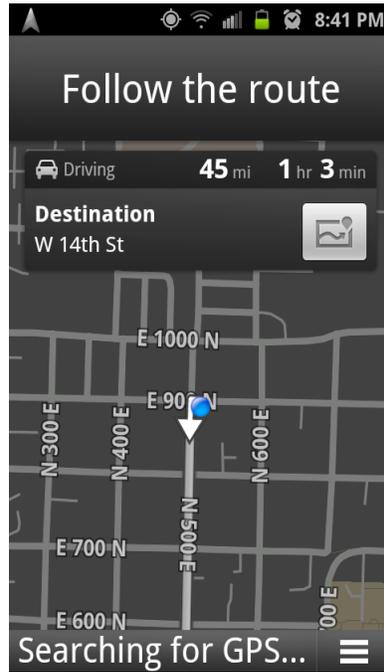
Call Button

The call button opens up the dialer filled with current shed's phone number as shown in the figure below.



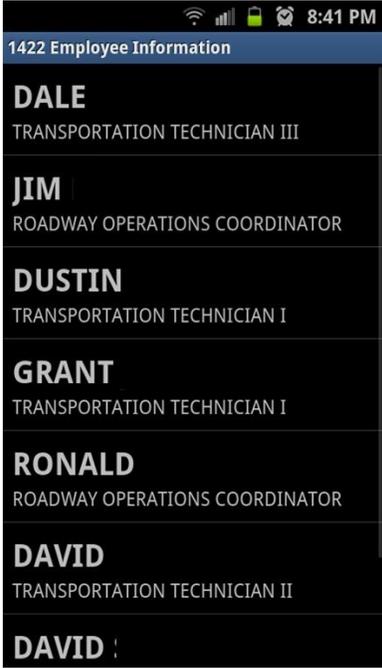
Direction

The direction button provides directions from the current user location to the shed selected as shown in the figure below.



Employees

The employee button provides information of the active employees and their designations of a particular shed as shown below.

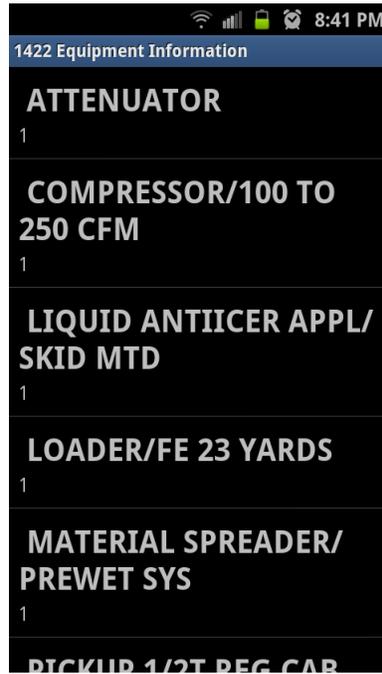


1422 Employee Information

DALE	TRANSPORTATION TECHNICIAN III
JIM	ROADWAY OPERATIONS COORDINATOR
DUSTIN	TRANSPORTATION TECHNICIAN I
GRANT	TRANSPORTATION TECHNICIAN I
RONALD	ROADWAY OPERATIONS COORDINATOR
DAVID	TRANSPORTATION TECHNICIAN II
DAVID	

Equipment

The Equipment button provides information about the active equipment and their quantity of a particular shed as shown below.



Appendix B : Sign Data Collection Application User Manual



Data Collection Application

Developed By:



Training Manual

Adapted from: http://www.google.com/intl/en/earth/outreach/tutorials/odk_collect.html

Data Collection Application

This manual is designed to provide survey enumerators with the knowledge necessary to successfully use an Android driven device to collect traffic sign related data. Upon opening the application, the main screen will display as shown below. This allows a number of selections to be made that will be explained in the following sections of this manual.

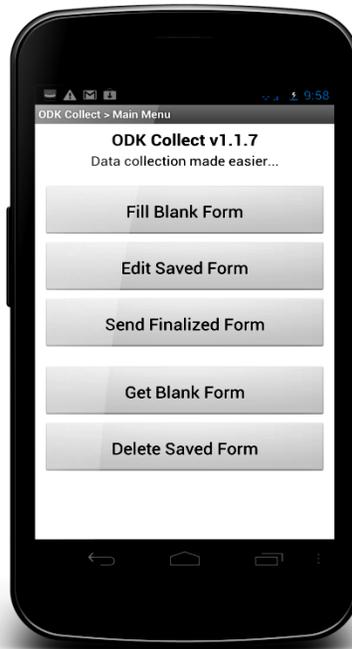
Installing ODK Collect

ODK Collect should be installed by a system administrator. If the application is installed correctly, a icon will show on the device as presented in the figure below.



Initializing the Application

Start the application by tapping the ODK Collect icon from either the home screen or the Android menu. You will see the home screen, which lists five options: “Fill Blank Form”, “Edit Saved Form”, “Send Finalized Form”, “Get Blank Form” and “Delete Saved Form”.



Getting a Blank Form

In order to fill out survey information on the Android device, you will need to make sure that you have the latest survey form downloaded from UDOT servers. Do this by making sure that your device is connected to the internet and tapping on the “Get Blank Form” button.

Toggle the form specific to your task and tap “Get Selected” to download the form for offline use. An example of this process is shown in the figure below.

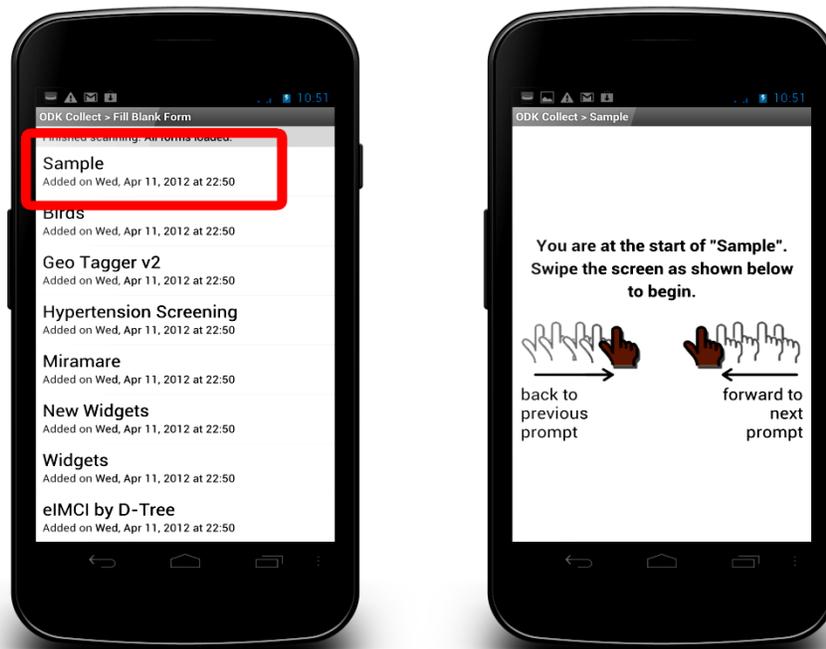


Fill a Blank Form

Once a form is downloaded to the device, it can be accessed by selecting the “Fill a Blank Form” button from the home screen of the application. Prior to doing this, turn off all unneeded device features, wifi for example. These features can be changed in the device settings accessed from the Android menu found on the device home screen.

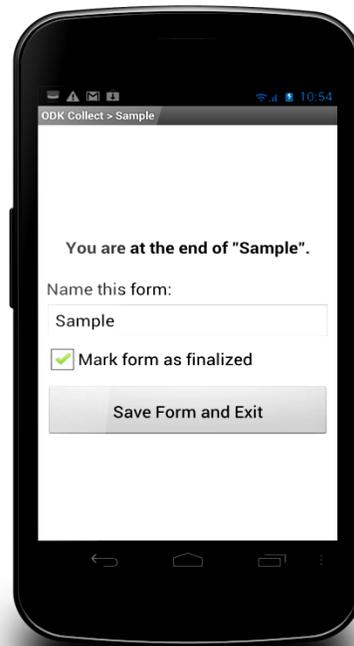
To begin a form:

1. Open the ODK Collect application on your Android device.
2. Select "Fill Blank Form."
3. Enter data into the form, swiping from right-to-left with your finger to get to the next questions. Enter data in all required fields.



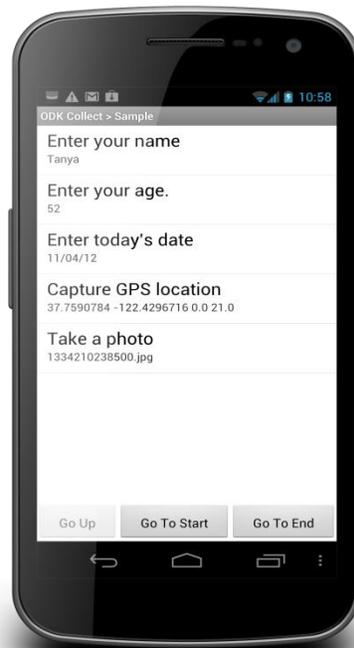
Completing a Form

When you are finished, make sure that the "Mark form as finalized" checkbox remains checked, and hit the "Save Form and Exit" button.



Reviewing a Form

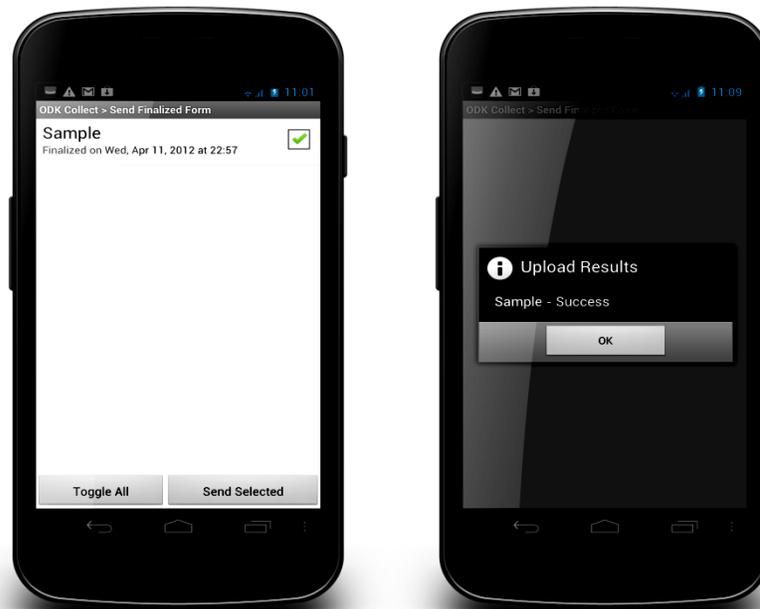
On the Home screen, you can review the data you collected, whether it has been saved as incomplete or saved as complete. Select "Edit Saved Form" and choose the data submission you would like to review. You can correct any mistakes or update the submission, opting to save your changes or ignore any changes made.



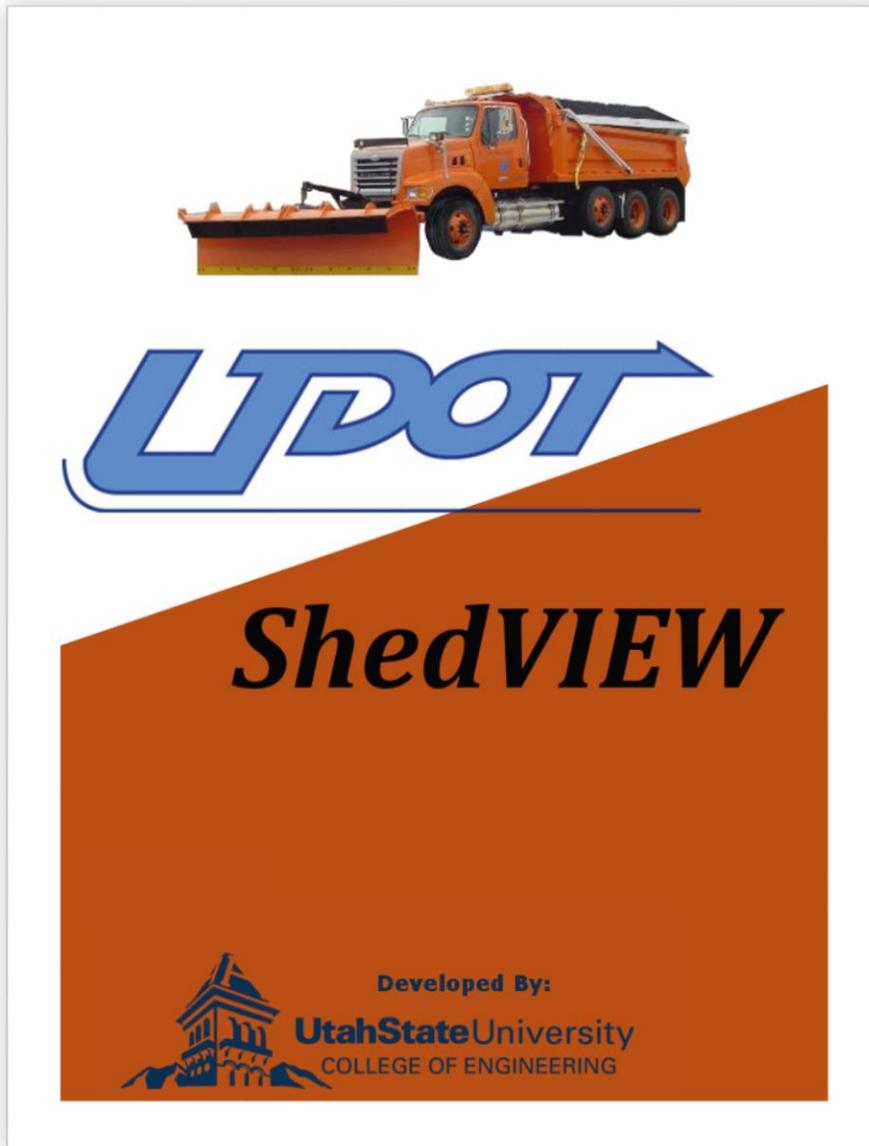
Sending a Finalized Form

Once you are done entering and reviewing data in the field, you are ready to send it to ODK Aggregate to upload it to UDOT servers. You must have an Internet connection to proceed with this step.

1. Make sure your device is accessing the Internet through the device wifi
2. Open ODK Collect and select "Send Finalized Form"
3. Check the box next to the entries that you wish to send. Those are the data submissions you entered in the field. The green check mark denotes selected files to be sent. You can select them individually or Toggle All.
4. Hit "Send Selected." Your files will send over the network to the UDOT server. You will see a message on your mobile device saying your data was sent successfully (or not, depending on your Internet connection).
5. If the send attempt is unsuccessful, continue trying to send until all form instances have been uploaded. You may have to wait for a better internet connection in the case of large uploads.



Appendix C : ShedVIEW Development Documentation



UDOT ShedVIEW

DESIGN DOCUMENT

ANDROIDMANIFEST.XML

This is the file where we define the properties for the project that we create. As of now we would support the application for the android device that Android OS 2.2 or above. We set this property using uses –sdk property as below.

```
<uses-sdk
    android:minSdkVersion="8"
    android:targetSdkVersion="8"
    android:maxSdkVersion="16" />
```

We define permissions required for the application in this file .For our application we require the below permissions

- GPS
- Location
- Internet

The above permissions are given as below.

```
<uses-feature android:name="android.hardware.location.gps"/>
<uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION"/>
<Uses-permission android:name="android.permission.INTERNET"/>
```

For every screen that we have to launch we create an activity and add an entry in this file. For the startup screen we define it as main activity as below.

```
<activity
    android:name=".UDOTSheildActivity"
    android:label="@string/app_name"
    android:screenOrientation="portrait" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category
android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

For other activities we define it as default in category section as below

```
<activity
    android:name=".StationInformation"
```

```

        android:label="@string/stationInformation"
        android:screenOrientation="portrait" >
        <intent-filter>
            <action
android:name="android.intent.action.StationInformation" />
            <category android:name="android.intent.category.DEFAULT" />
        </intent-filter>
    </activity>

```

Similarly we add up entries for all the screens that appear in our application.

Database:

File name: externalDB

Location: Assets folder in project.

Tables:

- *EMPLOYEES*: Contains the information about employees
- *EQUIPMENT*: Contains information about the equipment that each shed has.
- *LOCATION*: Contains information about sheds and their location.

Layout:

The screens that appear are xml files. We define the constituents of the screen in xml file and set the properties for individual component in xml file. These files are present at **res/layout** location in the project.

strings.xml

All the string constants used throughout this project would be present in strings.xml file present at res/values location in the project.

Images

The images used in the project are present in res folder. We have three resolutions for the images and images accordingly into their respective folders (drawable-hdpi, drawable-ldpi etc.).

DatabaseHelper

Prior to the starting screen of the application we define the wrapper class for the database applications. In our project or database wrapper class name is DatabaseHelper. We use the reference of this class throughout the project.

Variables:

<code>String DB_PATH =null;</code>	This variable is used to define the path of the database file
<code>private static String DB_NAME = "extenalDB";</code>	This variable is used to define the name of the database file.
<code>private SQLiteDatabase myDataBase;</code>	This variable is used to define the database.
<code>private final Context myContext;</code>	This variable is used to define the context.

Methods:

public DatabaseHelper (Context context): This method is used to define the context and also set the value of DB_PATH to the location of the database file.

public void createDataBase() throws IOException: This method is used to create the database. It checks if the database exists it does nothing. If the database does not exist it would get the reference of the readable database and tries to copy the database file to the DB_PATH location.

private boolean checkDataBase(): This method is used to check if the database exists or not at the DB_PATH location. If the database exists it returns true else false.

private void copyDataBase() throws IOException: This method is used to copy the database file from the assets folder to the DB_PATH byte by byte. If any error occurs while copying it throws up IOException.

public void openDataBase() throws SQLException: This method is used to open the existing database file.

public synchronized void close(): This method is used to close the database synchronously.

public void onCreate(SQLiteDatabase db):This method is used to define the functionality that needs to be done on the creation of the database.

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion): This method is used to define the functionality that needs to be done on the up gradation of the database.

public Cursor query(String table,String[] columns, String selection,String[] selectionArgs,String groupBy,String having,String orderBy): This is the most important method of this class with respect to our project. This method is used to query our database.

UDOTShieldActivity

This the screen that would be launched when the application starts. This class is associated with **main.xml** layout.

Variables: No variables are required in this class.

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together. It also defines the actions for the three buttons present in this layout i.e. **Auto, Manual and Location Data buttons.**

private void loadData(): This method is used to load the database from the external file to the DB_PATH location of the DataBaseHelper class.

public boolean onCreateOptionsMenu(Menu menu):This method is used to enable the menu to this screen.

public boolean onOptionsItemSelected(MenuItem item): This method is used to create the menu items for the screen.

ManualSearch

This is the screen called on the click of the Manual button on the initial screen of the application. This class is associate with **manual.xml**

Variables: No variables are required in this class.

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together. It also defines the actions for the two buttons present in this layout i.e. Region and City buttons.

private void loadData(): This method is used to load the database from the external file to the DB_PATH location of the DataBaseHelper class.

TestSearchList

This is the screen called on the click of Region button on the Manual search screen of the application. This class is associated with **regionlayout.xml**

Variables:

<code>private ListView lv;</code>	This variable is used as the reference for the list view.
<code>private static final String TAG = "RegionList";</code>	This variable is used for debugging.
<code>private static Cursor c;</code>	This variable is used to retrieve the region list from the database.
<code>private static DatabaseHelper myDbHelper ;</code>	This variable is used to perform database operations.
<code>ArrayList<String> regList = new ArrayList<String>();</code>	This list is used to represent the region list on the screen.

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together. It also defines the list view click event through the below code.

```
lv.setOnItemClickListener(new OnItemClickListener()
{
    public void onItemClick(AdapterView<?> parent, View child,int
    position, long id)
```

```

    {
        int passValue = regionInteger(((TextView)
        child).getText().toString());        Intent i1 = new
        Intent(TestSearchList.this, StationRegList.class);
            i1.putExtra("callIntent", passValue);

        startActivity(i1);

    }
});

```

private int regionInteger(String num): This method is used to return the region number which would be used by the list view item click event.

private void loadRegionList(): This method is used to load the distinct region numbers from the database to the regList arrayList.

StationRegList

This class is called on the list item click of the region screen. This screen would contain list of all the sheds along with their cities for a particular region. This class is associated with **stationreglist.xml**

Variables:

private static final String TAG = "StationRegLst";	This variable is used for debugging.
private static DatabaseHelper myDbHelper ;	This variable is used to perform database operations.
ArrayList<String> stationlst = new ArrayList<String>();	This variable is used to store station name and their city.
ArrayList<String> stationNum = new ArrayList<String>();	This variable is used to store only the station name
private ListView lv;	This variable is used as the reference for the list view.
private static Cursor c;	This variable is used to retrieve the

	station list from the database.
--	---------------------------------

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together. It also defines the list view click event through the below code.

```
lv.setOnItemClickListener(new OnItemClickListener()
{
    public void onItemClick(AdapterView<?> parent, View child,int
    position, long id)
    {
        Intent i1 = new Intent(StationRegList.this,
StationInformation.class);
        String passValue = stationName(((TextView)
child).getText().toString());
        i1.putExtra("callIntent", passValue);
        startActivity(i1);
    }
});
```

private String stationName(String myString)This method is used to return the station name which would be used by the list view item click event.

private void loadStationList(): This method is used to load the Station and City of a particular region into stationlst(station name and city combined) and stationNum(station name only).

StationInformation

This screen is used to display information of a particular station selected. This screen can be launched from many places (when we click on station name from any list view or from auto screen this screen would launched). This class is associated with **stationinfo.xml**

Variables:

<code>Button call_button = null;</code>	This variable is used to create a reference to call button.
<code>Button direction_button = null;</code>	This variable is used to create a reference to direction button.
<code>TextView phoneNUmber = null;</code>	This text view is used to display phone number of the station
<code>TextView siName =null;</code>	This text view is used to display the station name
<code>TextView superVisorName =null;</code>	This text view is used to display the supervisor name of the station
<code>Cursor c=null;</code>	This variable is used to retrieve the station information from the database.
<code>TextView stationAddress = null;</code>	This text view is used to display the station address
<code>TextView stationFaxNumber = null;</code>	This variable is used to display the station fax number
<code>private String location = null;</code>	This variable is used to store the coordinates of the station
<code>private String phonenumber = null;</code>	This variable is used to store the phone number of the station.
<code>private String passedValue;</code>	This variable is used to store the name of the station for which the information is being requested.

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together. This method initializes all the text views and buttons require. This method calls up the fillinformation method to retrieve the information from the database. This method also defines the functionality for the different button click event in the station information screen.

private void fillInformation(String stationName): This method is used to retrieve the information of station.

private void callButtonImplementation(): This method is used to define the call button functionality.

private void navigationImplementation(): This method is used to define the direction button functionality.

private void equipmentImplementation(): This method is used to define the equipment button functionality.

private void employmentImplementation(): This method is used to define the employee button functionality.

ManEquipment

This screen is called when we click on the equipment button in station information screen. It shows up the list of all the active equipment along with their count. This screen is associated with default **ListActivity** layout but uses **maneupitem.xml** layout to define the items in the list view.

Variables:

<code>private static final String TAG = "ManEquipment";</code>	This variable is used for debugging.
<code>private static Cursor c;</code>	This variable is used to retrieve the equipment list from the database.
<code>private static DatabaseHelper myDbHelper ;</code>	This variable is used to perform database operations.
<code>ArrayList<String> equipList = new ArrayList<String>();</code>	This variable is used to store the equipment list.
<code>ArrayList<String> stationNum = new ArrayList<String>();</code>	This variable is used as to store the station name.
<code>ArrayList<HashMap<String,String>> listm = new ArrayList<HashMap<String,String>>();</code>	This array list is linked up to the list view that would be shown.

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together. This method sets up the adapter to the listview.

private void loadEquipment(String myString): This method is used to retrieve all the active equipment along with their count for the station requested.

ManEmployee

This screen is called when we click on the employee button in station information screen. It shows up the list of all the active employees along with their designation. This screen is associated with default **ListActivity** layout but uses **manempitem.xml** layout to define the items in the list view.

Variables:

<code>private static final String TAG = "ManEmployee";</code>	This variable is used for debugging.
<code>private static Cursor c;</code>	This variable is used to retrieve the equipment list from the database.
<code>private static DatabaseHelper myDbHelper ;</code>	This variable is used to perform database operations.
<code>ArrayList<String> equipList = new ArrayList<String>();</code>	This variable is used to store the employee list.
<code>ArrayList<String> stationNum = new ArrayList<String>();</code>	This variable is used as to store the station name.
<code>ArrayList<HashMap<String,String>> listm = new ArrayList<HashMap<String,String>>();</code>	This array list is linked up to the list view that would be shown.

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together. This method sets up the adapter to the listview.

private void loadEquipment(String myString): This method is used to retrieve all the active employees along with their designation for the station requested.

SearchCityList

This screen would be launched when we click city button on the manual search screen. This screen shows the list of distinct cities from the database. This class is associated with **searchcity.xml**

Variables:

<code>private static final String TAG = " CitySearchList";</code>	This variable is used for debugging.
<code>private static DatabaseHelper myDbHelper ;</code>	This variable is used to perform database operations.
<code>private EditText et;</code>	This variable is used as the reference for the search box.
<code>ArrayList<String> cityList = new ArrayList<String>();</code>	This variable is used store the distinct city list.
<code>private ListView lv;</code>	This variable is used as the reference for the list view.
<code>private static Cursor c;</code>	This variable is used to retrieve the city list from the database.
<code>private ArrayList<String> array_sort= new ArrayList<String>();</code>	This array list is used store the sorted city list
<code>int textlength=0;</code>	This variable is used to check if anything is entered in the search box or not.

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together. It also defines the code to update the list according to the user input in the search box as below.

```

et.addTextChangedListener(new TextWatcher()
{
    public void afterTextChanged(Editable s)
    {
        // Abstract Method of TextWatcher Interface.
    }
    public void beforeTextChanged(CharSequence s,int start, int
count, int after)
    {
        // Abstract Method of TextWatcher Interface.
    }
    public void onTextChanged(CharSequence s,int start, int before,
int count)
    {
        textlength = et.getText().length();
        String mySearchString =
et.getText().toString().replaceAll("\n","");
        if(mySearchString.length()>0)
        {
            array_sort.clear();
            for (int i = 0; i < cityList.size(); i++)
            {
                if (textlength <= cityList.get(i).length())
                {
                    if(mySearchString.equalsIgnoreCase((String)cityList.get(i).subSeq
uence(0,textlength)))
                    {
                        array_sort.add(cityList.get(i));
                    }
                }
            }
            lv.setAdapter(new
ArrayAdapter<String>(SearchCityList.this,android.R.layout.simple_list_
item_1, array_sort));
        }
    }
}
else
{

```

```

        lv.setAdapter(new
        ArrayAdapter<String>(SearchCityList.this, android.R.layout.simple_list_
        item_1, cityList));
    }
    });

```

private void loadCityList(): This method is used to retrieve the distinct city list from the database.

StationCityList

This screen is launched on the click of the list view item in the search city screen. This screen shows up the list of the all the stations that are present in that city. This class is associated with **stationcitylist.xml**

Variables:

private static final String TAG = " StationregLst";	This variable is used for debugging.
private static DatabaseHelper myDbHelper ;	This variable is used to perform database operations.
private EditText et;	This variable is used as the reference for the search box.
ArrayList<String> stationLst = new ArrayList<String>();	This variable is used store the station list for a particular city.
private ListView lv;	This variable is used as the reference for the list view.
private static Cursor c;	This variable is used to retrieve the stations that belong to a particular city from the database.
private ArrayList<String> array_sort= new ArrayList<String>();	This array list is used store the sorted station list
	This variable is used to check if anything

<code>int textlength=0;</code>	is entered in the search box or not.
--------------------------------	--------------------------------------

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together. It also defines the code to update the list according to the user input in the search box as below.

```

et.addTextChangedListener(new TextWatcher()
{
    public void afterTextChanged(Editable s)
    {
        // Abstract Method of TextWatcher Interface.
    }
    public void beforeTextChanged(CharSequence s,int start, int
count, int after)
    {
        // Abstract Method of TextWatcher Interface.
    }
    public void onTextChanged(CharSequence s,int start, int before,
int count)
    {
        textlength = et.getText().length();
        String mySearchString =
et.getText().toString().replaceAll("\n","");
        if(mySearchString.length()>0)
        {
            array_sort.clear();
            for (int i = 0; i < stationlst.size(); i++)
            {
                if (textlength <= stationlst.get(i).length())
                {
                    if(mySearchString.equalsIgnoreCase((String)stationlst.get(i).subS
equence(0,textlength)))
                    {
                        array_sort.add(stationlst.get(i));
                    }
                }
            }
        }
    }
}

```

```

        lv.setAdapter(new
ArrayAdapter<String>(StationCityList.this,android.R.layout.simple_list
_item_1, array_sort));
    }
    else
    {
        lv.setAdapter(new
ArrayAdapter<String>(StationCityList.this,android.R.layout.simple
_list_item_1, stationlst));
    }

    });

```

private void loadStationList(): This method is used to retrieve the station list of a particular city.

private String stationName(String myString): This method is used to return the name of the station that is being clicked.

Auto View Implementation:

In the creation of the auto view for the map view I had used the external library. This library is used for the customized overlay. You can download the library using the link below.

<https://github.com/jgilfelt/android-mapviewballoons>

The process of including the library in project and its usage can be found at the site mentioned below.

<http://mobile.tutsplus.com/tutorials/android/android-sdk-build-a-mall-finder-app-points-of-interest/>

AutoTabLayout

This screen is launched on the click of the auto button on the initial screen of the application. The auto tab view consists of two views one for map view and another for list view. This view is associated with autotab.xml layout.

Variables: No variables are required in this class.

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together. It also defines components of the tab view that is MapView and ListView.

GMapsActivity

This screen is launched in the auto tab view screen when the Map tab is selected. This screen shows up the map view with your current location and also the five nearest stations. This view is associated with **mapautoview.xml**

Variables:

<code>private MapController mapController;</code>	This variable is used to reference the map controller
<code>private MapView mapView;</code>	This variable is used to reference the map view
<code>private LocationManager locationManager;</code>	This variable is used reference the location manager which would be used to obtain the current location.
<code>private GeoPoint currentPoint;</code>	This variable is used to store the current location's geopoint.
<code>private Location currentLocation = null;</code>	This variable is used to store the current location
<code>private MllOverlay currPos;</code>	This overlay is used to mark the current location
<code>private final int NUM_OF_STATIONS=5;</code>	The variable is used to store the number of nearest stations that needs to be shown
<code>private static final String TAG = "AutoMapView";</code>	This variable is used for debugging

<code>private static Cursor c;</code>	This variable is used to retrieve the stations and their coordinates from the database
<code>private static DatabaseHelper myDbHelper ;</code>	This variable is used to perform the database operations
<code>ArrayList<Station> mystatList = new ArrayList<Station>();</code>	This list is used to store the information of the stations that are nearest to it
<code>private int latSpan=14;</code>	This variable is used to control the zoom level
<code>private int longSpan=14;</code>	This variable is used to control the zoom level

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together. In this method we set the controls for the map view.

public void getLastLocation(): This method is used to fetch the coordinates of the last known location.

public void animateToCurrentLocation(): This method is used to set the animation to the current location.

public void centerToCurrentLocation(View view): This method is used to set the view in such a way that current location would be at center.

public String getBestProvider(): This method is used to obtain the best location provider.

public void setCurrentLocation(Location location): This method is used to set the current location based on the information provided by the provider.

public void drawCurrPositionOverlay(): This method is used to draw the overlay at the current position on the map view.

public void drawMalls(): This method is used to obtain the locations of the nearest stations.

protected boolean isRouteDisplayed(): This method would return false so that the route is not displayed.

public void onLocationChanged(): This method is called when the current location changes.

protected void onResume(): This method is called when the map view is resumed.

protected void onPause(): This method is called when map view is paused.

public void onProviderDisabled(): This method is used to show provider disabled message.

public void onProviderEnabled(): This method is used to show message when provider is enabled.

public void onStatusChanged(String arg0, int arg1, Bundle arg2): This method is used to display message when the status of the provider is changed.

private void dataFetch(): This method is used to fetch the information of the stations from the database.

private void displayList(): This method is used to draw the overlays at the nearest five stations with respect to the current location.

private void calculateSpan(): This method is used to calculate the zoom level for a map view.

private double distanceToCurrent(String coordinates): This method is used to return the distance from the current location to the coordinates requested.

Station

This class is used to create an object of station type. This class does not have any screen associated with it. This class is used in GMapsActivity class to store information of each station as objects.

Variables:

private String stationName ;	This variable is used to store the station name
private String city ;	This variable is used to store the city name of each station

private String coordinates ;	This variable is used to store coordinates of the station.
private double distance ;	This variable is used to store the distance of the station from the current location.

Methods:

public String getStationName(): This method is used to return the station name.

public void setStationName(String stationName): This method is used to set the station name.

public String getCity(): This method is used to return the city name.

public void setCity(String city): This method is used to set the city name.

public String getCoordinates(): This method is used to return the coordinates of the station.

public void setCoordinates(String coordinates): This method is used to set the coordinates for a station.

public double getDistance() : This method is used to get the distance of the station from the current location.

public double setDistance() : This method is used to set the distance of the station from the current location.

public Station(): This is the constructor for this class. This method is the first called when the object of station class is created. We have many constructors in this class depending up on the signature of the method that creates the constructor the corresponding method would be invoked.

public int compareTo(Station another): This method is used to sort the stations according to their distance from the current location.

MallOverlay

This class is used to show the Overlays on the map view in Auto screen. This class is used in GMapsActivity class.

Variables:

<code>private Context mContext;</code>	This variable is used to get the current context
<code>private ArrayList<OverlayItem> malls</code>	This variable is store the five nearest stations information
<code>private Location currentLocation;</code>	This variable is used to store the details of the current location.

Methods:

public MallOverlay(Drawable defaultMarker, MapView mapView): This method is the constructor for the class. This method would initialize the default marker on the map view.

protected OverlayItem createItem(int i): This method is used to create an overlay item.

public int size(): This method is used to used to give the number of overlays placed on the map view.

public void addOverlay(OverlayItem overlay): This method is used to add overlay item on the mapview.

public void setCurrentLocation(Location loc): This method is used to set the current location.

public Location convertGpToLoc(GeoPoint gp): This method is used to convert the location to geopoint.

protected boolean onBalloonTap(int index, OverlayItem item): This method is used to set the tooltip event for the overlay.

AutoListView

This screen is launched in the auto tab view screen when the List tab is selected. This screen shows up the list view five nearest stations. This view is associated with **autolistview.xml**

Variables:

<code>private LocationManager locationManager;</code>	This variable is used to get the location updates
<code>private static final long MINIMUM_DISTANCE_CHANGE_FOR_UPDATES</code>	This variable is used to set the minimum distance after which the location manager should refresh.
<code>private Location currentLocation;</code>	This variable is used to set the current location.
<code>private static final long MINIMUM_TIME_BETWEEN_UPDATES</code>	This variable is used to set the minimum time after which the location manager should refresh
<code>private final int NUM_OF_STATIONS</code>	This variable is used to set minimum number of stations that should be shown.
<code>private static final String TAG</code>	This variable is used for debugging.
<code>private static Cursor c;</code>	This variable is used to get the list of all the stations and their coordinates.
<code>private static DatabaseHelper myDbHelper ;</code>	This variable is used to perform database operations.
<code>ArrayList<HashMap<String,String>> listm</code>	This method is used to insert values into station array list
<code>ArrayList<Station> mystatList</code>	This variable is used to set the store the nearest five stations and their properties.

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together.

public void onItemClick(AdapterView parent, View view, int position, long id): This method is used to set the action for list item click event to show the station information.

private String stationName(int num): This method is used to return the station name.

private void dataFetch(): This method is used to get the list of all the stations and their properties from the database.

private void displayList(): This method is used to prepare the list of five nearest stations.

public String getBestProvider(): This method is used to specify the best provider to us.

private double distanceToCurrent(String coordinates): This method is used to calculate the distance between current location and the given coordinates.

MyLocationListener: This class used to get the location updates.

public void onLocationChanged(Location location): This method is used to set the action to be performed when the current location changes.

public void onStatusChanged(String s, int i, Bundle b): This method is used to notify us when the status of the provider changes.

public void onProviderDisabled(String s): This method is used to notify us when provider is disabled.

public void onProviderEnabled(String s): This method is used to notify us when provider is enabled.

EditPreferences

This screen is launched on the click of the menu options in the main screen i.e. UDOTShieldActivity screen.

Variables: No variables are required in this class.

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called.

protected void onResume(): This method is called when we resume the edit preferences screen.

SplashActivity

This screen is used to show the splash screen for the application. This screen is associated with splash.xml layout.

Variables:

private final int	This variable is used decide the time for
--------------------------	---

```
SPLASH_DISPLAY_LENGTH = 2000;
```

which the splash screen needs to be shown

Methods:

public void onCreate(Bundle savedInstanceState): This is the first method called in this class. This methods would link up layout and class together.\

protected void onResume(): This method is called when splash screen is resumed.

Appendix D : Sign Data Collection Application Development Documentation

This documentation covers the development efforts of the Sign Data Collection Application for use as reference in future endeavors. For clarity purposes, this documentation will be limited to include only development efforts that proved fruitful and led to the final product.

D.1 Collect

D.1.1 Programming ODK

Initial deployments of ODK included a Collect application module that remained unaltered from the original ODK releases. This proved useful to the development process as the ODK software collection is still under a highly active development phase. Final deployments meant for UDOT, however, should provide a customized interface and an update schedule that can be controlled by a system administrator.

Android application programming is performed using Eclipse, a java-based Integrated Development Environment (IDE). The first step in programming an application is to ensure a fully functioning version of Eclipse is available for use. This program can be downloaded and installed for free at <http://www.eclipse.org/downloads/packages/eclipse-mobile-developers/junior>. When first installing Eclipse onto a computer, care must be taken to ensure that all the correct steps (and there are many) are taken. Failure to ensure each of the steps is completed will result in the program throwing errors out when attempts to load applications are made. A detailed installation guide has been created by Google and can be found at <http://developer.android.com/sdk/installing.html>.

Prior to opening up the application for development in Eclipse the application must be cloned. This is accomplished using another program called TortoiseHG. This program is free and can be downloaded from the internet at <http://tortoisehg.bitbucket.org/download/index.html>.

To clone a program, TortoiseHG Workbench should be opened and Clone Repository selected from the File menu bar. This brings up a dialog box that requests the source and destination for the cloned application as shown in Figure D.1. The Source is provided by the ODK developers and the most recent release resides at <https://code.google.com/p/opendatakit.collect/>. The destination should reside in an easy to find location within a folder created for the cloned files. Once the Source and Destination fields are filled out, the Clone button will copy all of the required files into the folder selected in the set-up process.

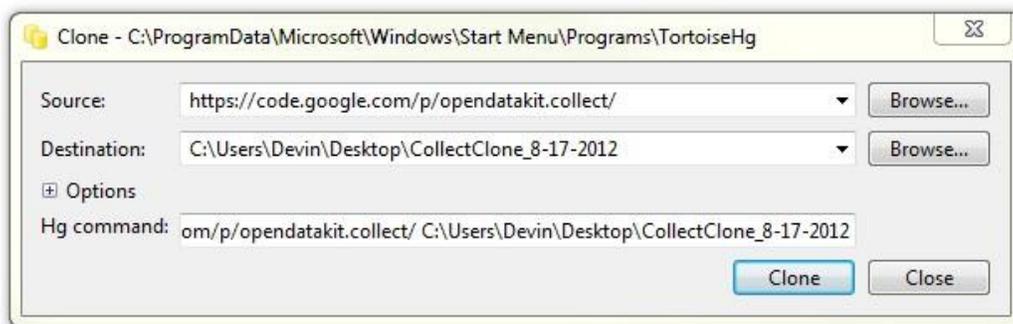


FIGURE D.1 Cloning an Application

Once the cloning process is complete, the application files need to be imported into the Eclipse IDE. This is done by opening Eclipse and selecting Import from the File drop down from the menu bar. Once the Import dialog box opens, the selection “Existing Projects into Workspace” should be selected. Clicking on the next button in this dialog will allow the user to select the clone repository created by TortoiseHG using a Browse button. Once selected, the application will show under the Projects window. Ensure that the project is selected and click the Finish button.

Once imported, some effort may be required for error resolution in order to get an application to build correctly. This process can be accomplished through internet searches related

to the various error codes thrown by eclipse. Once error free, changes can be made to the application as needed. There are two categories of changes that can be made to the stock Collect application. The first change modifies only visual data and does not have an effect on the overall programming codes. The second type of change requires programming knowledge and adds or removes functionality from the application.

In testing efforts, only visual modifications were made to the Collect application. These changes primarily occurred within the program strings and the startup screen. The strings (res/values/strings.xml) under the collect interface control how the program displays text. Changing these values is one of the easiest ways to customize the application. The strings listed in Table D.1 are a good place to start when beginning customization activities. To further modify the application, changes can be made to the splash screen (res/layout/splash_screen.xml). This will customize the application to UDOT while still referencing the application's ODK heritage as per the license terms.

TABLE D.1 Suggested String Changes

Name	ODK Value
app_name	ODK Collect
app_url	http://opendatakit.org
change_username	ODK Username
click_to_web	Tap to visit http://opendatakit.org
default_server_url	https://opendatakit.appspot.com
main_menu_details	Data collection made easier
open_data_kit	Open Data Kit
powered_by_odk	Powered by Open Data Kit
server_platform_odk_aggregate	ODK Aggregate

No functionality was added or removed in initial testing efforts, but there is allowance for such changes to occur within the ODK system. To make these changes, knowledge in java

programming is necessary to maintain program integrity and functionality. Future additions that would be beneficial in the case of sign management include the incorporation of a Compass application into the survey, similar to the way that barcode scanning is incorporated. Additionally, autocomplete selection capability would be beneficial, especially in the case of MUTCD code selection. In all programming efforts, it is suggested that the ODK forums be referenced to ensure that the changes suggested are not already planned, as is the case with the above two suggestions.

Once all required changes are made to the application, it can be exported as a .apk file in Eclipse by selecting Export from the File menu bar and following through the prompts. The .apk file then can be loaded onto an Android device and installed through a file explorer application such as Dropbox. This installation requires special permission in the security settings to allow for applications from Unknown Sources, something that will be prompted for upon an installation attempt if not already allowed.

Distribution of custom Collect deployments is straightforward. To accomplish this, the .apk file needs to be uploaded to a web server. Once accessible, mobile users are given a link to the application's location through either an e-mail or a QR code. Clicking on this link with their mobile devices will download the application and install it onto the Android device.

D.1.2 Licensing Restrictions

The ODK software is licensed under the Apache Foundation 2.0 license and can be modified or changed as needed so long as the requirements of its open-sourced license are met. The license, according to the Apache License and Distribution FAQ (48), is described in layman's terms in the bullets below:

It allows you to:

- Freely download and use Apache software, in whole or in part, for personal, company internal, or commercial purposes;
- Use Apache software in packages or distributions that you create.

It forbids you to:

- Redistribute any piece of Apache-originated software without proper attribution;
- Use any marks owned by The Apache Software Foundation in any way that might state or imply that the Foundation endorses your distribution;
- Use any marks owned by The Apache Software Foundation in any way that might state or imply that you created the Apache software in question.

It requires you to:

- Include a copy of the license in any redistribution you may make that includes Apache software;
- Provide clear attribution to The Apache Software Foundation for any distributions that include Apache software.
- It does not require you to:
- Include the source of the Apache software itself, or of any modifications you may have made to it, in any redistribution you may assemble that includes it;
- Submit changes that you make to the software back to the Apache Software Foundation (though such feedback *is* encouraged).

The full license can be found at <http://www.apache.org/licenses/LICENSE-2.0.html> and it should be noted that it is the final authority on the license terms. The descriptions above, while informative, hold no legal authority.

D.1.3 Reference Web Sites

ODK is still being developed and tweaked to provide new features and remove bugs and other problem areas within the coding. As such, the software components are considered living and should be kept up to date by an administrator in the case of any custom deployment. The developers of ODK maintain a series of web sites listed in Table D.2 that can be used to obtain latest updates and commentary.

TABLE D.2 Applicable Web Sites

Purpose	Site Address
ODK Main Web-Site	http://opendatakit.org/
ODK Question Forum	https://groups.google.com/forum/?fromgroups#!forum/opendatakit
ODK Developer Forum	https://groups.google.com/forum/?fromgroups#!forum/opendatakit-developers
ODK Developer Site	http://code.google.com/p/opendatakit/
ODK Downloads	http://code.google.com/p/opendatakit/downloads/list
ODK Release Notes	http://code.google.com/p/opendatakit/w/list
Form Building Site	http://formhub.org/syntax/

D.2 Aggregate

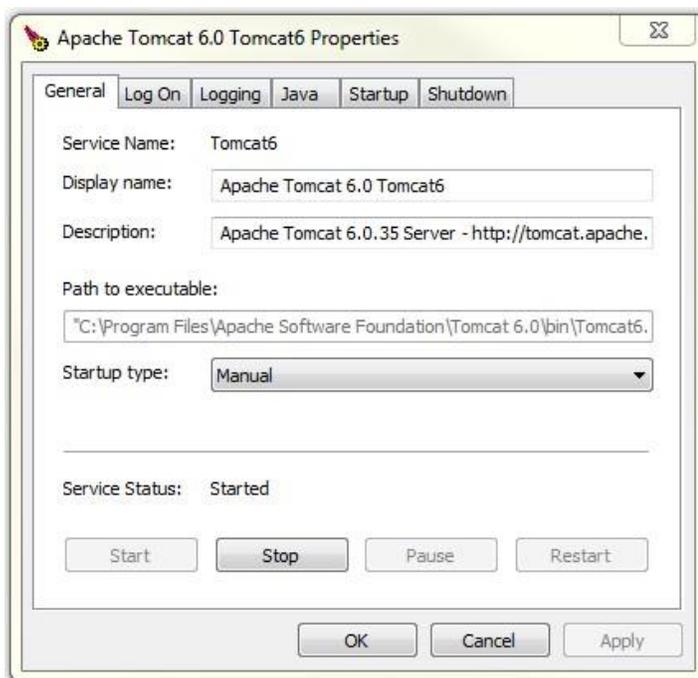
Aggregate is used to wirelessly gather submissions from multiple devices running collect and combine them into a single .csv file. This file can then be incorporated into a central database for further analysis. The following paragraphs discuss specific issues relating to the set up and proper control of Aggregate. If additional help is needed, detailed information is available on the Aggregate site under the question mark icon in the top left corner.

D.2.1 Deployment

For testing purposes, Aggregate was installed on a TIMELab desktop computer at one of the research stations. Deployment was straightforward and was accomplished by following the directions provided by ODK developers that can be found at <http://opendatakit.org/use/aggregate/tomcat-install/>. Final deployment should occur on continuously available computer located either on the USU campus or in UDOT headquarters. This computer can be multi-purpose and does not have to be a dedicated server as would sit on a rack in a back room.

In the event that the Aggregate host computer is restarted, it may not come online automatically. There are two possible fixes to solve this problem. The offline issue lies in the way that Tomcat Server interacts with Windows 7. The server needs administrator privileges to run and without them it will not start up correctly. To temporarily manually restart the server, first search for the file Monitor Tomcat in the Start Menu search bar in Windows 7. Once the program is found right click on it and select Run as Administrator from the drop down menu. Under Service Status in the opened window select Start to begin the service as shown in Figure D.2 and then select OK to close the program. Once this process is completed, Aggregate can be accessed as normal.

A permanent fix requires navigation to the C:\Program Files\Apache Software Foundation\Tomcat 6.0\bin and right clicking on the Tomcat6w.exe file. Click on the Properties menu item and then choose the Compatibility tab. At the bottom of that dialog check the box next to Run this program as an administrator. If the option is selectable, change the same settings after clicking the Change settings for all users button. Click the OK button to close the dialog and the changes will be effective immediately. This will eliminate any further occurrences of restart errors permanently.

FIGURE D.2 Monitor Tomcat Dialog

D.2.2 Connecting Collect Devices to Aggregate Instance

Once a custom deployment is up and running, devices need to be pointed to the newly created web server and permissions need to be granted. This is an easy process but involves a number of intricate details that can be frustrating if not accounted for.

D.2.2.1 Setting up Aggregate for Submissions

Aggregate must be set up in a way that devices running collect can log onto the server and pull or submit data from it. To do this, navigate to the Aggregate directory established during the Tomcat and Aggregate installation. First time visitors to this site will be presented with a screen that asks them to sign in with an Aggregate password, a Google account, or for Anonymous Access. To make changes, the administrator who set up Aggregate for the first time needs to sign in using the same Google account provided in the installation process. Once signed

in, the administrator can allow access via the other two tabs in the Site Admin tab at the top of the Aggregate web-page.

Upon navigation to the Site Admin tab, ensure that the Permissions page is toggled and then navigate to the Add Users section at the bottom of the page. This section is where permission for users is given to view and access data. In the blank box type in a user name that will be used by devices running Collect to make submissions (“Collect” or “Input” for example) and then select the Add button. This will add the user to the list of users found below the Edit Users section. Once the user has been added, permissions need to be set up to allow for submissions. On the line that has been created for the new user, the Username should be as defined in the Add Users section. The Full Name section should be left blank. Click on the Change Password button and choose a password that can be easily input on an Android keyboard (not a lot of alternating numbers or symbols and letters). Leave the Account Type as ODK and leave all the checkboxes blank except for the box under Data Collector. Once the changes are complete, select the Save Changes button. Once saved, Aggregate is ready to accept submission.

D.2.2.2 Pointing ODK Collect to Aggregate Deployment

Each device that will be running Collect needs to be set up in a way that submissions can be sent to and forms pulled from Aggregate. The steps to pointing ODK Collect to a local deployment of Aggregate are as follows:

4. Open Collect on an Android device
5. Select the menu from the home screen
6. Ensure the Server Platform remains ODK Aggregate

7. Change the URL to the custom URL established during Aggregate and Tomcat setup. This will be a modified version of the URL given in the browser window because:

- ODK requires http:// to precede the URL
- The URL truncates after the “:port#/ODKAggregate” for ODK

An address that would be acceptable within ODK therefore would follow the format as shown in the two examples below:

- http://computername.network:port#/ODKAggregate
- http://A00000000-3.bluezone.ueu.edu:8080/ODKAggregate

8. Enter the Username defined in Aggregate under section D.2.2.1
9. Enter the Password defined in Aggregate under section D.2.2.1
10. Leave all other settings as their default and press the back button on the device to return to the main menu.

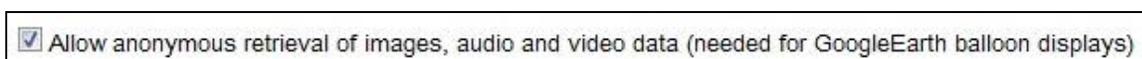
The connection settings can be tested by choosing the Get Blank Form button from the Collect menu, ensuring first that the device is connected to the internet. If all settings are correct a page containing a list of forms or a statement relating to the lack of forms present on the Aggregate server should appear. If the connection is not successful, the first place to begin troubleshooting is with the Aggregate URL in the settings menu.

D.2.3 Multiple Users and Data Retrieval from Aggregate

In order for users to access Aggregate information, an account must be set up for them by the Aggregate site administrator. This is accomplished on the Site Admin tab of Aggregate. Any number of users can be added on this page and various permissions granted. The permissions granted vary and specific access points will have to be set up as required by UDOT.

Regardless of the number of permissions set up, however, the box next to the option to Allow anonymous retrieval of images, audio and video data (needed for Google Earth balloon displays) MUST be checked as shown in Figure D.3. This will allow access to media files once the data is exported.

FIGURE D.3 Aggregate Retrieval Setting



D.2.4 Uploading Forms

Survey forms can be uploaded on the Form Management tab by selecting the Add New Form button at the top-right of the web page. This will connect to the local computer and search for Forms and Optional Media Files (See discussion in Section D.3). Navigate on the local computer to the .xml form file to be uploaded and press the Upload Form button. Once the form is uploaded, it will appear on the Form Management page. Ensure that the boxes under Downloadable and Accept Submissions are checked so that Collect users can access and use the forms correctly.

D.2.5 Exporting Data

Data can be exported under the Submissions tab. The Submissions tab contains a listing of all data instances that have been submitted via systems running Collect. To be usable in database programs, this data must be exported into a usable file. To do this, first make sure that the Filter Submissions page is activated and then select an option from the Form drop down menu at the top left hand side of the web page. Once the correct Form data is loaded and shown, click the Export button at the top right of the web page. Leave the default Type as CSV file and

Filter as none and click on the Export button. Aggregate will then move to the Exported Submissions section of the Submissions tab and generate a CSV file. The generation may take a short amount of time to finish. Once complete the file link can be selected to download the CSV file to the local computer.

D.2.6 Future Work

The process of setting up devices with Collect and connecting to Aggregate is fairly straightforward. Future work should focus on programming a solution to automatically export submitted data as CSV files and then incorporating that export directly into the central database. By providing this type of automation, UDOT will not have to task an employee to specifically manage Aggregate, thus limiting the possibility of a corrupt database. Given this solution, data will be able to seamlessly transfer from data collection devices running Collect directly into the central database.

The end goal of this transfer is to maintain a single database table that contains all instances of recorded sign information within an agency. As such, the main table should be formatted identically to the output of the Collect application with inapplicable fields being left blank or null. When developing a database solution for UDOT, care should be taken in the handling of duplicate or updated record instances within the main database. Ideally a unique numerical identifier should be assigned to each instance of signage within an agency. This unique identifier, however, will be separate from an additional unique identifier required by the database. This is better described using an example presented in Table D.3. In formatting the database in this way, duplicate sign entries can easily be identified and removed from the table, the analysis, or both as required within an individual agency. This formatting style will ensure database reliability as well as viability in long term applications.

TABLE D.3 Identifier Example

Unique Instance #	Unique Sign #	Record Date (M/D/Y)	Retroreflectivity Value
11422355	1	7/12/05	42
11422356	2	7/12/05	283
11422400	1	7/12/12	32

D.3 Form Builder

Survey forms for incorporation into Collect can be effectively created using a number of different methods including directly in XML, in an ODK tool called Build, or in a Microsoft Excel file. The latter of these options has been found to be the easiest of these methods in trials where initial building as well as later modification was tested. The key in programming these forms is identifying the correct programming syntax to ensure compatibility with ODK.

D.3.1 Form Template

The easiest way to get started with form building is by using an ODK supplied template that details all of the widgets and items available for use in a form deployment. It is available at http://opendatakit.org/wp-content/uploads/2012/05/sample_xlsform.xls. The template is fairly straight forward and provides a good example to use in programming efforts.

D.3.2 Syntax

The syntax for building a form can be found at <http://formhub.org/syntax/>. This syntax will aid in the more technical aspects of form development as well as supplement the form template for developers. While generally useful, this syntax document is not updated as often as the ODK platform and as a result, key improvements to form programming and widget

implementation can be overlooked. It is recommended that ODK blog postings be followed closely to identify new and useful form features.

D.3.3 Converting Excel Form to XML

Prior to the form being usable in Collect, conversion must take place through which an XML file is created from the Microsoft Excel form. This conversion requires all syntax and programming within Excel to be correct. It also requires the Excel form to be in .xls format and not .xlsx as is default in the newer versions of Excel. ODK provides an online conversion tool to both check the programming and convert the form at <http://opendatakit.org/use/xlsform/>. The process of converting the form, finding errors, addressing errors, and reconversion can take time. Patience through this conversion process is crucial to making sure the form functions correctly within Collect. If the form is programmed incorrectly, it will cause the Collect application to crash which can be frustrating to the form user.

D.3.4 General Notes

The following bullets contain notes that pertain to the current release of Collect which is (v. 1.2 1013) as it relates to building a form.

- The AutoComplete functionality has been disabled due to a bug in version 1.1.7. The estimated time until arrival of a fix is unknown.
- Version 1.2 1013 allows Collect to access an external app to obtain a string value. This can be accomplished using a path defined as ex:intentpath on the appearance attribute of xls form. A sample form using this functionality can be found at <http://code.google.com/p/opendatakit/source/browse?repo=androidextras#hg%2FBreathCounter>

- Version 1.2.0 allows Cascading Selects which are explained in detail on the ODK web site.

Appendix E : UDOT Sign Management OMS Fields

The following output was generated using the UDOT Operations Management System. It was obtained through personal communication with Mike Marz, an UDOT Information Technology Analyst. The output describes each of the sign attributes that are currently being collected by UDOT personnel.

E.1 Element 410 – Sign Support

There are two tables involved with this feature, Sign Support Inventory & Sign Face Inventory. Each Sign Support entry will have at least one associated Sign Face entry associated with it.

When a user first enters the Sign Inventory database, he or she will see the Sign Support table. This table has all the data associated with the support of a sign. The Sign Face table has all the data associated with the sign face itself. After selecting a support from the Sign Support Inventory table, one can click the Sign Face Inventory tab to see all the sign face records associated with that support. This way we can show multiple sign faces with one support.

TABLE E.1 Sign Support Database Elements

OMS Screen Description and EXCEL Column Heading	Data Type	Acceptable Pick List Values	Unit of measure or Description	Format	Example
Element	Number	410		Three digits	410
Route	Alpha / Numeric		LRS Route Number	four digits and one	0015P
Direction	Pick List	Both (all) NEG POS		Characters	Both (all)
Offset	Number		Distance, in feet, from the edge of pavement to the centerline of the sign. No value is entered for overhead signs.	Two digits	14
Start MP	Number		Starting	999.999	123.589

			Milepoint in miles		
Number of Signs	Number		The number of individual sign signs attached to the sign support	One integer	2
Side Designation	Pick List	LEFT RIGHT		Characters	LEFT
Sign Position	Pick List	CENTER LEFT MEDIAN OVERHEAD RIGHT T INTERSECTION UNKNOWN		Characters	RIGHT
Sign Support Type	Pick List	DELINEATOR POST DOUBLE POST FIVE POST FOUR POST GATE ONE POST OTHER POST IN BARREL SIGN BRIDGE SIGNAL POLE SPAN WIRE STRUCTURE TRIPLE POST UTILITY POLE		Characters	ONE POST
Sign Support Material Type	Pick List	ALUMINUM CONCRETE FIBERGLASS PLASTIC SLIP BASE STEEL UNKNOWN WOOD		Characters	STEEL
Sign Support Size	Pick List	2" CHANNEL 2X2 3-1/2X6 I-BEAM 3-3/8X10 I-BEAM 3X4 I-BEAM 4X10 I-BEAM 4X12 I-BEAM 4X4 4X6 4X8 I-BEAM 5X5 6X10 I-BEAM 6X6 6X8 8X8 I-BEAM 10"		Characters	6X8

		I-BEAM 12" I-BEAM 14" I-BEAM 3" I-BEAM 4" I-BEAM 6" I-BEAM 8" OTHER SQUARE TUBE 2" SQUARE TUBE 4" SQUARE TUBE 6" SQUARE TUBE 8" TUBE 2" TUBE 3" TUBE 4" TUBE 5" TUBE 6" TUBE 8" TUBE 12" TUBE 18" TUBE 24"			
Sign Base	Pick List	Asphalt Casing Concrete NONE Sleeve Soil		Characters	
Station	Pick List	See allowable pick list values in Table 18.		Characters	2431 - West Jordan
District	Pick List	1411 - Region 1 Maint Admin 2411 - Region 2 Maint Admin 3411 - Region 4 Maint Admin 4411 - Region 4 Maint Admin		Characters	2411 - Region 2 Maint Admin
Administrative Unit	Pick List	8301 - Maintenance Admin	8301 - Maintenance Admin	Characters	8301 - Maintenance Admin
X Coordinate	number		UTM Zone 12 X-Coordinate	999999.999	423836.280
Y Coordinate	number		UTM Zone 12 Y-Coordinate	9999999.999	4503300.720
Signs Class Code	Pick List	410 - MFI Signs		Characters	410 - MFI Signs
Comments	Character		Pertinent comments concerning the fence	200 char max	Black PVC coated chain link
Date Updated	Date		Date the field data was gathered	Mm/dd/yyyy	06/15/2010

E.2 Element 410 – Sign Face

Notes:

- Table E-1 is a dependant (child) table of Table E-2
- Each Separate Sign face in the sign installation will be entered as a separate line item
- Standard Manual of Uniform Traffic Control Devices (MUTCD) or *UDOT Sign Manual* sign designations will be used
- Signs not identifiable by MUTCD or Utah Sign Manual Designation will be identified by the Numeral 1 followed by the background color of the sign in all capital letters. (i.e. 1BROWN, 1WHITE, 1BLUE, and so forth).
- At least one color digital photograph, not to exceed 200 KB in .JPEG format, of freeway guide sign installations, excepting MUTCD number E1-5, E5-1, D10-1, D10-2, D10-3, and D5-series, will accompany the submittal. Sign file name will be as follows: Route, Direction, Milepoint, Side Designation, and Sign Number in the installation.

TABLE E.2 Sign Face Data Elements

OMS Screen Description and EXCEL Column Heading	Data Type	Acceptable Pick List Values	Unit of measure or Description	Format	Example
Sign ID	Number		A unique integer	Six numbers	1
Sign Visibility	Pick List	Clear Obscured by Buildings Obscured by Curve Obscured by Vegetation		Characters	Clear
MUTCD Codes	Pick List	Standard MUTCD sign designations or UDOT Sign Manual designations or 1BLACK 1BLUE 1BROWN 1GREEN 1ORANGE 1WHITE 1YELLOW		Characters	R1-1
Sign Orientation	Pick List	East North North East North West South South East South West Unknown West	The direction that the retroreflective face of the sign faces	Characters	North West
Sign Legend	Character		The text legend as it appears on the sign. Example: "Ogden 7", "Sanpete County". Leave this field blank for any sign that contains a legend element in the sign name. Speed limit signs need not have the text "SPEED LIMIT" recorded	Characters	Provo 35
Sign Width	Number		The horizontal width of the sign face in inches. For diamond-shaped signs, the measurement along an edge of the sign in inches;	Three digits	72

			For hexagonal and round signs, the diameter in inches		
Sign Length	Number		The vertical measurement of the sign face, in inches. For diamond-shaped signs, the measurement along the edge perpendicular to the sign width, in inches; Leave blank for hexagonal and round signs	Three digits	36
Height	Number		The height in feet from the center/bottom edge of the sign to the nearest pavement surface	Two digits	7
Sign Face Material	Pick List	I - Engineer Grade II - Super Engineer Grade III - High Intensity III, IV - High Intensity Prismatic III, IV, X - High Intensity Prismatic IV, VIII - Crystal Grade IX - Diamond or Omni-View VII, VIII, X - Diamond Grade LDP VIII - MVP Prismatic X - Crystal Grade XI - Diamond Grade DG3		Characters	III - High Intensity
Sign Backing Material	Pick List	Aluminum Other Plastic Steel Wood		Characters	Wood
Sign Illumination	Pick List	BACKLIT BUTTONS FLASHER ILLUMINATED NONE OTHER		Characters	NONE
Comments	Character			200 char max	Obsolete Adopt-a-highway sign
Date Updated	Date		Date the field data was gathered	Mm/dd /yyyy	06/15/2010