

Polling with Mobile Devices

Benefits and New Possibilities

Denis Andrašec

Polling with Mobile Devices

Benefits and New Possibilities

Master's Thesis

at

Graz University of Technology

submitted by

Denis Andrašec

Institute for Information Systems and Computer Media (IICM),
Graz University of Technology
A-8010 Graz, Austria

Evolaris Next Level GmbH
A-8010 Graz, Austria

November 2013

© Copyright 2013 by Denis Andrašec

Advisor: Assoc.Prof. Dipl.-Ing. Dr.techn. Martin Ebner
Advisor Evolaris: Dipl.-Ing. Thomas Ebner



Umfragen auf mobilen Endgeräten

Mehrwert, Potential, Möglichkeiten

Diplomarbeit

an der

Technischen Universität Graz

vorgelegt von

Denis Andrašec

Institut für Informationssysteme und Computer Medien (IICM),
Technische Universität Graz
A-8010 Graz, Österreich

Evolaris Next Level GmbH
A-8010 Graz, Österreich

November 2013

© Copyright 2013, Denis Andrašec

Diese Arbeit ist in englischer Sprache verfasst.

Begutachter: Assoc. Prof. Dipl.-Ing. Dr.techn. Martin Ebner

Betreuer Evolaris: Dipl.-Ing. Thomas Ebner



Abstract

Mobile devices have become ubiquitous. They offer unique capabilities and technologies like multitouch displays, portability, location via GPS, wireless internet connection, and ease of use. Exploring those features in the context of a research field has the potential to offer new possibilities and innovation. This thesis aimed to do this by examining the use of mobile devices as a tool for researchers in the social sciences, for both qualitative as well as quantitative methods. The development process for such a tool, called Tablet Computer Assisted Personal Interviewing (TCAPI), is the main focus of this work.

A literature survey was done to show possibilities of mobile devices in the context of interviews, surveys, polling, and questionnaires. Following this, a survey of applications in this field was conducted for Android and iOS operating systems. With these results and the concept of the TCAPI project, a prototype application with a large feature set was developed, and its potential was evaluated through qualitative interviews.

The literature survey, as well as the comparison of existing applications revealed that there is a limited offering of applications which foster mixed method approaches in empirical social sciences. Also, applications are often tied to web services, making them a companion and not a full featured tool. The interviews revealed that researchers found the tool useful and saw large potential for innovation through the features that were discussed and developed during the course of this work.

Kurzfassung

Mobile Endgeräte sind allgegenwärtig. Sie bieten einzigartige Möglichkeiten und Technologien, wie Multitouch Displays, Portabilität, Standortbestimmung durch GPS, drahtlose Internetverbindung und einfache Handhabung. Diese Eigenschaften im Kontext eines Forschungsgebietes haben das Potential neue Möglichkeiten und Innovationen hervor zu bringen. Diese Arbeit versucht dies zu unterstützen indem mobile Endgeräte als Werkzeug für Forscher in sozialen Wissenschaften untersucht werden. Dies soll für qualitative als auch quantitative Methoden betrachtet werden. Die Entwicklung eines solchen Werkzeuges, mit dem Namen Tablet Computer Assisted Personal Interviewing (TCAPI), ist der zentrale Fokus dieser Arbeit.

Eine Literaturstudie wurde durchgeführt um Möglichkeiten von mobilen Endgeräten im Kontext von Interviews, Umfragen und Fragebögen zu zeigen. Darauf folgt ein Vergleich von Applikationen in diesem Feld für die Betriebssysteme Android und iOS. Mit diesen Ergebnissen und dem Konzept des TCAPI Projektes wurde ein Prototyp mit einer Vielzahl von Funktionalitäten entwickelt und dessen Potential mit Hilfe qualitativer Interviews geprüft.

Die Literaturrecherche und der Vergleich von vorhandenen Applikation zeigte ein limitiertes Angebot von Applikationen welche das Mischen von qualitativen und quantitativen Methoden der empirischen Sozialforschung ermöglichen. Des Weiteren waren diese oft an Web-Services gebunden und erfüllten daher nur eine Funktion als Zusatz, nicht aber als eigenständiges, umfangreiches Werkzeug. Die Interviews zeigten, dass Forschende den Prototypen als nützlich sahen und dass viel Potential an Innovation in den besprochenen und entwickelten Funktionen zu finden ist.



This work was created in cooperation with Evolaris Next Level GmbH.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Place

Date

Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Ort

Datum

Unterschrift

Contents

Contents	ii
List of Figures	iii
List of Tables	v
List of Listings	vii
Acknowledgements	ix
Credits	xi
1 Introduction	1
2 Introduction to Mobile Devices and Development	3
2.1 Mobile Devices	3
2.2 Development: Testing and Agile	5
3 Introduction to Empirical Social Sciences	7
3.1 Goal and Research Process	7
3.2 History	9
3.3 Qualitative vs. Quantitative Methods	10
3.4 Interviews	10
3.5 Mixing Modes	11
4 Technology and Mobile Devices in the Context of Empirical Social Science	13
4.1 Web	13
4.2 Computer Assisted Personal Interviewing	14
4.3 Technologies of Mobile Devices	15
4.4 Surveys with Mobile Devices	18
5 Comparison of Interview and Questionnaire Applications on Mobile Devices	19
5.1 Approach to Evaluation and Comparison	19
5.2 Applications	20
5.3 Comparison and Results	31

6	TCAPI: A New Tool for Interviews and Surveys	35
6.1	Requirements	35
6.2	Features	37
6.3	Selected Features	42
6.4	Implementation	43
6.5	Interviews - Field Study	50
7	TCAPI: Development Process and Implementation Details	63
7.1	Agile Development Process	63
7.2	Test Driven Development	66
7.3	Development Environment	67
7.4	Frameworks	68
7.5	queXML	76
8	Conclusion and Outlook	79
8.1	Conclusion	79
8.2	Outlook	80
A	TCAPI User Stories	81
B	TCAPI Mockups	85
	List of Abbreviations	89
	Bibliography	91

List of Figures

3.1	The Research Process	8
4.1	Nokia 770	16
5.1	Loop Survey Maker	22
5.2	TabSurvey	22
5.3	DataField	23
5.4	SF-36 Survey	25
5.5	WiseHunch	29
5.6	FluidSurveys	30
6.1	The TCAPI Ecosystem	44
6.2	TCAPI Database Model	46
6.3	TCAPI Prototype Home	46
6.4	TCAPI Prototype Import	47
6.5	TCAPI Prototype Questionnaire	48
6.6	TCAPI Prototype Interview	48
6.7	TCAPI Prototype Full Screen Mode	50
6.8	TCAPI Prototype Export	51
6.9	Questions Array, Mask	52
6.10	Questions Multiple Selection, Single Selection, Text	53
6.11	Feature Question QF1	58
7.1	Waterfall Model	64
7.2	Scrum	65
7.3	CoreData Example	71

List of Tables

5.1	Comparison: Survey Features	31
5.2	Comparison: Question types	33
5.3	Comparison: Import/Export	34
6.1	Selected Features for Implementation	42
6.2	Supported Questions	51
6.3	Interview evaluation question QI01	55
6.4	Interview evaluation question QI02	55
6.5	Interview evaluation questions QI04-QI10	56
6.6	Feature Questions	57
7.1	Audio Session Categories	73
7.2	iOS XML Parser Performance	75

Listings

7.1	Creating and deleting a student within an context.	70
7.2	Fetching students.	71
7.3	Dropbox Upload Class.	74
7.4	XML structure of a simple questionnaire.	76
7.5	A simple XML schema.	76

Acknowledgements

First, I would like to thank my girlfriend Karin, who supported me the whole time, giving advice when needed and encouraging words when things got rough.

I want to thank my advisor, Martin Ebner for the help and the much needed direction that he gave me during this last part of my studies. Further more, I want to thank my advisor at Evolaris, Thomas Ebner who trusted me with the work. Additionally, big thanks goes out to Rafael Schögler and Martin Griesbacher. Thanks for all the support and advice during development.

I also want to thank all my fiends, with whom the endless weekends spend at the university were so much more enjoyable. Those moments will always be remembered fondly.

Finally, I am grateful for the help, advice, and understanding that I received from my parents, my brother and his wife. Thank you for always being there for me.

Denis Andrašec
Graz, Austria, November 2013

Credits

I would like to thank the following individuals and organizations for permission to use their material:

- The thesis was written using Keith Andrews' skeleton thesis [Andrews, 2012].
- The data from the qualitative interviews is used with kind permission from Rafael Schögl and Martin Griesbacher.

Chapter 1

Introduction

“Begin at the beginning and go on till you come to the end; then stop.”

[Lewis Carroll, Alice in Wonderland]

Mobile devices have become ubiquitous, and they should also be considered as serious research tools. Another field of study that possibly could benefit from the special properties of mobile devices are empirical social sciences. The methods to gather data in this field range from qualitative methods, like interviews, to quantitative methods, like standardized questionnaires. This work aims at exploring possibilities and offering new approaches. This is done by an extensive literature survey as well as a comparison of applications for interviews, questionnaires and surveys.

A large area of possible innovation is the use of mobile devices to foster mixed method approaches, which are the focus of the Tablet Computer Assisted Personal Interviewing project and the accompanying mobile application. This project, which was under the scientific administration of Prof. Manfred Prisching, was conceived and organized by Rafael Schögler and Martin Griesbacher at the Centre of Social Research at the Karl-Franzens-University of Graz. The software development was done in cooperation with Evolaris Next Level GmbH. Literature findings, results of an application comparison, and the initial project plan were used as an input for development of a prototype application. This process is described in this work. The potential of the prototype is evaluated through interviews with qualitative and quantitative approaches.

Chapter 2 is a introduction to mobile devices and development. After this, chapter 3 introduces the reader to empirical social studies and qualitative as well as quantitative methods. Chapter 4 takes a look at the technologization of interviews and surveys and at mobile devices in that context. After this follows an evaluation and comparison of mobile applications in chapter 5. Building on the findings of the previous chapters and the initial project design, the prototype application is conceived, described, and evaluated in chapter 6. Chapter 7 concerns itself with details of the development process and implementation. Finally, chapter 8 serves as the conclusion and outlook to this work.

The overall question that this work examines is: "How can a useful research tool for both qualitative and quantitative research methods be implemented on mobile devices and what are relevant and innovative features in this context?"

Chapter 2

Introduction to Mobile Devices and Development

“Always design a thing by considering it in its next larger context - a chair in a room, a room in a house, a house in an environment, an environment in a city plan.”

[Eliel Saarinen, Finnish architect, 1873–1950.]

This chapter is concerned with the basic definitions of words and concepts of mobile devices and the process used for prototype development. It serves as a quick overview over the key topics that will get introduced in more detail in later chapters. Section 2.1 explains what falls into the definition of mobile devices, respectively as mobile devices as defined for the purposes of this thesis. And section 2.2 briefly describes important concepts of the development process.

2.1 Mobile Devices

Mobile devices have become ubiquitous in daily lives. Those are smartphones like the Google Nexus 4, tablets like the Apple iPad and similar devices from various hardware manufacturers. These devices have a modern mobile Operating System (OS), to which Android from Google, iOS from Apple, Windows 8 and Windows Phone 8 from Microsoft is counted. Those are the top operating systems and the top two are looked at more specifically throughout this work. Others are Symbian from Nokia and BlackberryOS from RIM.

All those devices share certain characteristics, which sets them apart from other mobile devices like handhelds and Personal Digital Assistant (PDA), and those are like already mentioned, their operating systems, the touchscreen technology, the form factor and location capabilities. While other devices may offer some of those technologies, the combination of them is what makes modern mobile devices stand out, and their success with consumers is a testimony to this.

2.1.1 Touchscreen

Touchscreen technology, where a user can manipulate software with the touch of his fingers, has improved to the point where multiple points and touches are registered. Called multi touch,

it is part of most modern smartphones and tablets. It enables users to manipulate content on their devices through gestures. Some of those are swipes, taps, double-taps, pinch-to-zoom and press and hold. Figure shows a graphical representation of this. These are paradigms which are very different from those found on desktop computers where the mouse and keyboards are the main input devices, fundamentally changing the dynamics of user interaction [*Human Interface Guidelines* 2013]. Therefore, interaction models for software need to be adapted when design applications for mobile devices. There are more possibilities, but also numerous drawbacks [Ebner, Stickel, and Kolbitsch, 2010]. For example, they do not have the precision of a pointing device, making it necessary to create larger touch targets which are suited for finger input. Balancing this needs is a challenge with the constrained screen size often found on mobile devices, due to their smaller form factor.

2.1.2 Form Factor

Mobile devices are, as their name suggests, very portable. The screen sizes are typically between 3.5 and 5 inches for smartphones, and 7 to 10 inches for tablets computers. Both device classes offer long battery life and small weight. Computing power, albeit constantly rising, is still smaller when compared against their desktop and Personal Computer (PC) counterparts, therefore limiting their possible applications. Also, using larger computing resources negatively affects battery life, making it essential to constrain Central Processing Unit (CPU) usage if this is a concern. When designing a user interface for a device with a small screen, it is not enough to shrink everything down. A different approach is needed, or as Dawson et al. [2012] puts it, "Boil the application down to the most critical functions and content, and then lay them out strategically in the available screen space. For example, action buttons should go in the lower third of the screen, where they are most easily tappable."

2.1.3 Location

Mobile devices offer mechanisms to determine location, which is the key component for offering Location Based Services (LBS). Different technologies can be used to determine the location of a device. For example, most smartphones and a rising number of tablets offer Global Positioning System (GPS) capabilities. This is possible if the device receives radio signals from at least three GPS satellites, calculating the position with signal timings, strength and triangulation. Those signals are weak, and therefore don't work within buildings. Another approach is triangulation with the cell towers or wireless network [Dutzler, Ebner, and Brandner, 2013]. Also, combining those technologies offers more accuracy [Vaughan-Nichols, 2009-02]. The conclusion is, if a device offers one of these wireless technologies, it can estimate the position rather accurately, making LBS possible. Examples of these services are mapping, directions, nearby recommendations, and social networks. This feature of mobile devices is seen as the one with almost unlimited potential, and [Vaughan-Nichols, 2009-02] concludes that demand for LBS will strongly increase if the technology is adapted wide enough and good business models are found.

LBS can also change users behavior in their neighborhoods. Foursquare for example lets users check-in at points of interest, with the incentive to collect points and get coupons and deals in the locations. While this is leading people to explore more around them, this type of "volunteering" the own location gives private organizations extensive information about their

users, essentially enabling them to create detailed profiles which lead them to anticipate behavior [Wilson, 2012]. While essentially all social networks collect information about their users, organizations can gain even more detailed information when LBS come into play.

2.2 Development: Testing and Agile

Software development is inherently complex, and has become even more so over the course of time. With large projects, changes during development are the norm, not the exception, making it nearly impossible to account for every detail up front. Throughout a project, multiple things have to be contemplated. Data structures, database designs, computation complexities, compositions, user interface and interaction, source code management, and distribution are only some of the important key points, and numerous others need to be considered.

During prototype development, two approaches, which go hand in hand, have been incorporated to cope with the difficulties of software development. One is an agile development process, where software is continuously iterated upon in short circles, and the other is Test Driven Development (TDD). In the latter, tests are written first, and implementation code afterwards. By following both, software development becomes more dynamic, in the sense that unpredicted changes are easier react upon, fostered by the short iterations and the security that tests provide. Especially in prototype development, where large changes are the norm, these approaches are indispensable.

Chapter 3

Introduction to Empirical Social Sciences

“To write it, it took three months; to conceive it three minutes; to collect the data in it all my life.”

[F. Scott Fitzgerald, American author, 1896–1940.]

This chapter is a short introduction to empirical social studies. A look at the history of this field will provide insights into how this form of research came to be, and why it is so valuable. The research process is explained, as well as modes and methods of interviews and questionnaires.

3.1 Goal and Research Process

Schnell, Hill, and Esser [2011, Page 24] cites that the goals of empirical social sciences are, practically, to ensure that humankind can live a more rational and humane live through problem solving and, theoretically, scientists want to create an accurate and proof-able model of reality. Therefore, social sciences and its methodologies can be seen as a tool to verify theories and models. Examples for those are human behavior, interaction, or social rules. It must be possible to deduct empirical consequences from theories, and it must be possible to prove them true or false with the techniques that empirical social sciences offer. Surely, this can not be done naively, and a framework needs to support these aspirations. Following these intentions, is the research process described by Schnell, Hill, and Esser [2011, Page 26] as a sequence of different steps as seen in Figure 3.1. These are described below:

Selection of the Research Problem Choosing a research problem, or object of study depends on the current research in the professional literature. A problem should be chosen which can be evaluated objectively, and the results and properties must be empirically verifiable.

Building a Theory A theory for a given object of study is either available, there are similar objects of study from which a theory can be adapted, or a new theory has to be built with the theoretical principles and techniques found in social sciences. Either way, to build a good theory, extensive literature research is crucial.

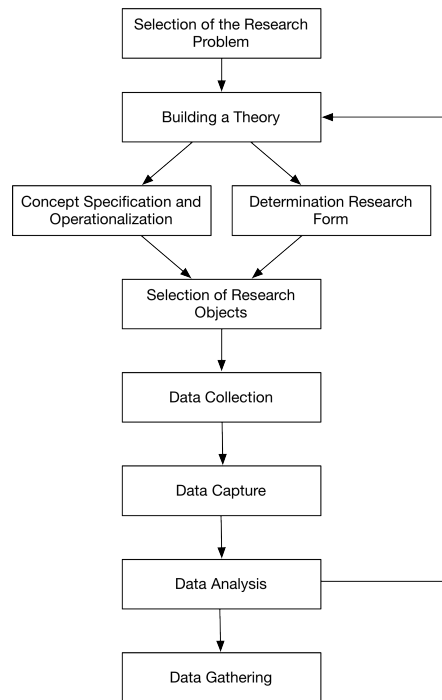


Figure 3.1: The research process adapted from Schnell, Hill, and Esser [2011, Page 26].

Concept Specification and Operationalization The concepts and terms of theories have to be specified very accurately, so that everyone understands what exactly the meaning of them is. In the operationalization step, the terms have to be clear, measurable, and distinguishable. Therefore, they are measurable and comparable to each other. Questionnaires, questions and other various tools can be used to collect those measurements.

Determination of the Research Form The research form defines how and when measurements are taken, how many measurements are taken per person, or if measurements are taken for groups of people. Choosing the research form also depends on funding and time, and several approaches can be taken to cope with such restrictions.

Selection of Research Objects Strong considerations should go into how many objects of a given population are chosen for research purposes. Very rarely it is possible to conduct research on the whole population of a country for a poll about the next elections. In this case, a representative set of research objects of the whole population has to be chosen. This process can be very tedious, because of bureaucracy and privacy concerns, but is very efficient when done right. For example, polls about election results are often based on the data that was collected from a small percentage of the population. Nonetheless, they are often very close to the actual results.

Data Collection Data collection is done through interviews or observations. When choosing the interview approach, questionnaires have to be designed, printed and interviewers have to be trained to use them accordingly.

Data Capture The data has to be captured in a way that it can be analyzed later. Capturing can be through audio recordings, writing or even digital if the data collection part is computer assisted. Once captured, the data has to be saved in a structured format. This step is

called 'Coding'. Coding and cleaning the data of errors can consume much time, energy and resources.

Data Analysis In this phase of the research process, and often the data is analyzed using statistical procedures supported by computer programs. Also in this phase, there is a feedback loop to the research theory. The better the theory is the simpler the data analysis becomes. Therefore, it can be important to revise the theory in the wake of new and unexpected data.

Publication Ultimately the outcomes of the research have to be published to contribute to its associated field of research. This is done either through a book or journal article. In the latter case, the article is sent to the publisher, where it is peer-reviewed. After this, the article usually has to be revised to be published.

As already mentioned during the descriptions of the phases, computer assisted techniques can be used for data collection, capture, and analysis, which is described throughout this work.

3.2 History

The history of social sciences begins with a history of data gathering. Each society with bureaucracy needs data about its population. It is used for administration, collecting taxes, and selecting people for war duty. Early societies in China, Greece, Persia and the Roman Empire already collected data and statistics about agriculture and trade.

King William the Conqueror commanded several studies about the estates of the feudal lords, after the Norman Conquest of England in the 11th-century, because of economic reasons. Information about the size and value of estates, number of residents and their positions, and the size of fields and woodland was collected. Although there are written results, they are not in tabular form, making quantification difficult. This early form of data collection and statistics is regarded as a predecessor of empirical social science. Similar forms of data collection efforts could be found through the second half of the 17th century in France and Prussia [Schnell, Hill, and Esser, 2011].

Atteslander [2010] describes that by the 19th century, there were not only attempts to describe the nature and the material world, but there was also a need to, accurately and scientifically, describe cultural and social phenomena. These new demands quantified too many things in social areas, even if there is no theoretical necessity. For example, Quetelet used this approach in his book "Social Physic", released in 1835. He proposed, that collecting a large number of observations, leads to the discovery of structures and general rules about social life. By systematic collection and quantification of all factors of life, he wanted to find regularities and patterns. Nevertheless, even if this approach finds regularities and patterns, it is not evident that valuable knowledge about society follows from them, making most of them irrelevant.

Following this thinking, Comte, also understood sociology, a term that he first used, as a scientific field of study. Different from Quetelet, he proposed that a data collection effort has to be guided by a theory, which serves as an explanation of eventual conclusions. However, both approaches could fail to deliver universally acceptable theories about social circumstances. Because by only quantifying observations and social interactions, those become too abstract and theoretical. According to the findings of French social theorist Pierre Guillaume Frédéric Le Play, qualitative methods are a solution for this problem. He conducted direct observations of

working class families. Whereas Quetelet gathered data to find patterns of society as a whole, Le Play tried to confirm his theories of individual cases and only collected information that he found to be relevant for his case.

Qualitative approaches gained more and more importance after the second world war in the United States of America. Also, in the 1960s Germany there were discussions when qualitative approaches or quantitative measurements are the right approach for a given field of study. An important work of this time is the book, "Die Integration qualitativer und quantitativer Methoden in der empirischen Sozialforschung - Theoretische Grundlagen und methodologische Konzepte", by Kelle [2008], which describes both approaches. Today, both approaches are widely accepted and commonly used tools, even in combination, and they interact with each other over the whole research process.

3.3 Qualitative vs. Quantitative Methods

Qualitative and quantitative methods are both an integral part of empirical social sciences today. They are not exclusively used and can be combined, differing in the researchers role and level of standardized questions.

In qualitative approaches, the researcher asks questions, usually in the form of a face-to-face interview. Those are very open, and subjective impressions of the researcher are desired to gain more insights. The researcher asks people in a certain context, with the goal to gain insight by letting them answer very freely. Therefore, this form of approach is used when there is not much available information about a field of research and theories or hypothesis have to be build at first, making it an explorative approach. A fixed hypothesis prior to qualitative interviews is not seen as desirable because the interviewer would take the research in a predetermined direction, limiting the extractable knowledge. Qualitative research was usually seen as a preliminary stage of quantitative research, but Atteslander [2010, Page 24–25] argues that qualitative research does provide insights on its own, although they are different from those in representative studies.

Contrary to qualitative approaches, subjective observations of the researcher are undesired in quantitative research. The data collection is very standardized, and collected data is statistically analyzed and compared. This approach is about quantifying information and variables. The order of questions is in large parts predetermined, and it is desirable to have identical circumstances in every situation where the data is gathered. Standardized approaches are usually used if there already exists a strong hypothesis or theory and they serve to prove or disprove it. [*Qualitative Forschung - Ein Handbuch*, Pages 24–25]

3.4 Interviews

In recent years, additionally to face-to-face and mail surveys, research forms like telephone interviews or web questionnaires gained significance. These developments have to be taken into account when discussing interviews and questionnaires. Interviews for research purposes are fundamentally different from those that take place in everyday life. To illustrate this Atteslander [2010, Page 110] describes three primary properties of common interviews:

1. They are a social process between two or more participants.

2. They are goal oriented. In every interview exists an interest.
3. They are defined through the used medium and the surrounding environment. The former is the language and its underlying cultural norms and rules and the latter are things like location, time, or other, not directly involved, people.

While interviews, which satisfy scientific criteria, are also goal oriented, the main differentiator is that they are continually controlled with the guidance of a theory. Through this approach, it can be understood how particular environmental conditions influence collected data. Further more, Atteslander [2010, Page 134] divides interviews in slightly structured, partly structured, and strongly structured.

In slightly structured interviews, the interviewer typically has no written questions. He or she can lead the conversation in different directions, appropriate it to the interviewee, ask for more details, or vary the wording of questions. Even though the interviewer has goals, the interview process follows the answers of the interviewee, creating a conversation that is lead by those answers. The priority is on exploring the interviewees field of experience.

On the other scale of the spectrum lies the strongly structured interview. Prior to the interviews, a questionnaire is created, defining content, number of questions and the order of questions. Therefore, interviewers should strictly abide by the structure of the questionnaire, as wording and usage of answer categories is predetermined. Reasonable interviews should last between 30-60 minutes and are also dependent on the number and difficulty of the questions in this timeframe. Strongly structured interviews are almost always preceded by slightly or partly structured interviews.

In-between those approaches are partly structured interviews. Questions are predetermined, but the order can be varied. Also, the interviewer has the possibility to ask more questions about a topic the interviewee speaks about, and ask more detailed questions according to the answer.

Written questionnaires that are answered without an interviewer are often used where monetary limitations exist. In contrast to spoken interviews, there is no regulation and control. In most cases, it is not possible to monitor when, how, and with whom the questionnaire was filled out.

3.5 Mixing Modes

As already hinted, there are several different modes of data collection. Not only do they differ in questionnaire design, with open questions in qualitative methods, and standardized ones on quantitative method, but also by the medium. While early efforts focused on personal face-to-face interviews and mail interviews, today a broad array of additional modes is possible. They are telephone interviews, web surveys, Computer Assisted Personal Interviewing (CAPI), Computer Assisted Telephone Interviewing (CATI), as well as Computer Assisted Self-administered Interviewing (CASI), to name a few. Furthermore, mixing them together has become more and more popular, which can have its own advantages and disadvantages.

According to Leeuw [2005], researchers are faced with the questions which modes to combine to harvest the best results for their research question, both in data quality as well as economically, or more exactly put, Leeuw [2005] points out, "When designing a survey the goal is to optimize data collection procedures and reduce total survey error within the available time and budget. In other words, it is a question of finding the best affordable method, and sometimes

the best affordable method is a mixed-mode design. Survey designers choose a mixed-mode approach because mixing modes gives an opportunity to compensate for the weaknesses of each individual mode at affordable cost."

For example, large advantages can be achieved when mixing CAPI with CASI approaches on sensitive topics. The interviewer is letting the interviewee fill out personal and sensitive information autonomically, because of the higher perceived privacy, harvesting better data for those questions.

Evidently, mixing modes can produce desirable results. Mixing modes when only one is used for data collection is considered unproblematic, when the purpose of the others are reminders or pre notifications for surveys. However, if multiple modes are used for data collection, comparability of the data must be achieved. Therefore, careful design is needed, and mode effects should be known and taken into consideration [Roberts, 2007].

Chapter 4

Technology and Mobile Devices in the Context of Empirical Social Science

“To raise new questions, new possibilities, to regard old problems from a new angle, requires creative imagination and marks real advance in science.”

[Albert Einstein]

In this chapter shows how computer technology has affected interviews and surveys. Two approaches are described. Section 4.1 offers a short look at web surveys and its problems, and after that examples of computer assisted personal interviews are given in section 4.2. Following this, technologies of mobile devices are looked at in the context of interviews and questionnaires in section 4.3. Finally, section 4.4 discusses current literature findings about questionnaires on mobile devices.

4.1 Web

Conducting surveys over the web has several positive properties. Examples are lower cost of delivery and almost real time data collection. Participants also noted that they liked this form of surveys because it is easy to use. However, surveys conducted on the web have lower response rates than mail surveys. Manfreda et al. [2008] conducted a meta-analysis where they found, that web response rates are about 11% lower than those of mail surveys. Another prime example for problems in web surveys is representativeness. The U.S. Census Bureau releases reports on internet penetration for the USA. In 2009, the population that had internet access at home was 68.7% and rose to 71.1% in 2011 [*Computer and Internet Use in the United States 2013*]. Dillman et al., 2009 argues that the 2009 numbers are not enough for general public surveys. Cook, Heath, and Thompson [2000] also argue, that if almost all of the population is captured, the sample must be representative. Since it is not possible to reach the whole population through web surveys, and response rates are almost never 100% anyway, a part of the population has to be selected which is representative of the whole. This is more relevant than the response rate as noted by CookCook, Heath, and Thompson [2000]. As an example, they present that election polls with 1% of the population are, most of the time, highly representable of the whole population. If there is a uncertainty of the responses, a high response rate would still be desirable. Nonetheless, there have to be ways to reduce those uncertainties.

Another problem is nonresponse bias. This occurs when the population which is not part of a survey, or did not respond to it, would answer the questions significantly different than those that are. The mail survey is very prone to this problem. If this problem occurs, the survey is *not* representative of the whole population. According to the literature research done by Armstrong and Overton [1977], respondents who answer later on surveys are more similar to non-respondents. Therefore, they can be used to extrapolate nonresponse bias. Statistical significance on reducing error with extrapolation was reached [Armstrong and Overton, 1977, Page 7]. There is one problem as it is not possible to measure the time from the respondents awareness of the questionnaire or survey until its completion [Armstrong and Overton, 1977, Page 2]. This is a datapoint that could be collected on mobile devices.

4.2 Computer Assisted Personal Interviewing

According to Caviglia-Harris et al. [2012] approaches where Computer Assisted Personal Interviewing was successfully used date back to the 1980s, and they are increasingly becoming the standard in interview situations. They found the following concerns when using such approaches, "Potential concerns examined in the literature include technical feasibility, respondent acceptance, interviewer acceptance, effects on data quality, and cost." In CASI approaches, feasibility and respondent acceptance was already found to be met, and this can also increasingly be seen in interview situations. One of the most prominent requirement for feasibility in interviews is weight and portability. Interviewers have to be able to easily transport the computer equipment and, depending on the situation, should easily use the device in sitting or standing positions [DeBree et al., 2008; Fann et al., 2009; Larsson, 2006].

In their research, Caviglia-Harris et al. [2012] were focused on testing the feasibility of CAPI methods in remote areas with the help of ruggedized laptop computers with touch-screens and GPS capabilities. In developing countries, face-to-face interviews are used to gather information about population and development dynamics, and traditionally those things are done with Paper Assisted Personal Interviewing (PAPI) approaches. With the computerized approach, several advantages could be observed:

Data Quality The quality of the data was increased as lower error rates were reported because computer questionnaires offer possibilities for skips, required questions, and data consistency checks. For example, fields that only allow numerical inputs within predefined ranges could be implemented.

Reduced Travel Time With the help of GPS equipped devices, right locations in remote areas were easier to find, therefore, strongly reducing travel time. Prior to this, directions had to be found from neighbors.

Greatly Reduced Data Entry The times needed for data entry were greatly reduced.

Easy Meta Data Collection With an CAPI approach, it is easy to collect metadata like interview time, duration and location.

However, Caviglia-Harris et al. [2012] noted that there were also significant difficulties because of the usability of computer systems, noting that programming or changing the questionnaires needed deep knowledge about the system and also programming skills, calling for

trained personnel on site to perform those tasks if short term changes to questions were needed. The design of CAPI methods seems to be the most influential factor when comparing the collected data to the one obtained with other modes. For example, Fuchs, Couper, and SE [2000] found that interview duration is the same with PAPI and CATI approaches when questions are structured the same way. Concluding, that where computer assisted techniques are slower, this can be attributed to questionnaire design, rather than to the technology.

4.3 Technologies of Mobile Devices

4.3.1 Touch Screen

When performing questionnaires on touch-screen devices, it has to be determined if the method is feasible in different contexts or usage scenarios and if the quality of data is comparable to other methods, like pen-and-paper. If those two are met, all the advantages of computer administered methods come into effect. Those are almost error free data, cost savings, and immediate availability of the gathered data for analysis. First let's look at how touch-screen devices were used to date for questionnaires and how they stacked up against other methodologies. The benefits and drawbacks are pointed out by examining the literature. Moreover, essential properties of those new methodologies are to be found in the literature, and we shall also see what they are.

Several studies have been conducted on the feasibility of touch-screen computers for filling out questionnaires and forms. A key point of examination is if the gathered data is comparable to classic methodologies like pen-and-paper questionnaires. Of course, it is difficult to show the feasibility of those new approaches in general because of the context where they are used, and exposure to the technology of participating people has to be taken into consideration.

One field of interest is the improvement of Health Related Quality of Life (HRQOL). The use of touch screen computers in such an environment was examined by DeBree et al. [2008]. The goal is to capture information about health of patients over a longer period of time. Classic approaches of data collection, like pen-and-paper, are difficult in such a setting because of missing resources for questionnaire distribution, data collection and analysis, or put more bluntly, it is a matter of cost. To tackle this problem, an approach where patients autonomously answered questions on a touch-screen computer on their visits to the hospital was tested. Most patients found the system easy to use, and compliance to fill out questionnaires on later visits was 67%. This seems low, but it could be increased by training the hospital personal accordingly. The collected data was available in real-time for analysis through a database.

Similarly, Fann et al. [2009] tried to show the feasibility of touch-screen equipped computers for depression screening of cancer patients. As a questionnaire, the Patient Health Questionnaire 9, or PHQ-9 was selected. It is very short with only nine items and is especially useful for detecting changes and treatments effects. Therefore, it can be used for longitudinal screenings. Those take place over a long period of time, and require a screening method that is both reliable and not cost intensive. Patients filled out the questionnaire in a waiting area, and the time to completion was usually around two minutes. Older patients needed longer, which is attributable to the lack of experience with computers. The computerized method of data collection was feasible in this environment.

So using only a touch-screen computer in such a difficult setting was shown to be feasible.



Figure 4.1: Nokia 770 Hand-Held.

More important is showing that this method is also comparable to well known and tested methods. That is precisely what Larsson [2006] examined. He compared a touch-screen computer assisted method to the classical form of data collection through pen-and-paper, also in a clinical setting. Patients were asked to rate the quality of care they were given during cancer treatment. One half of patients performed the questionnaire with the touch-screen, the other half used paper. It was shown that the majority of ratings were comparable between the two modes. Patients seemed to prefer the touch-screen version. The quality of data was excellent as no errors in the data were reported. The questionnaire was designed in a way that *all* questions have to be answered. It is not possible to move on with a question if it is not answered, but it is possible to go back and change answers. This has the effect that no data was missing, in contrast to the paper questionnaire where 14% of patients did not provide one or more answers, even though they were instructed to answer all questions. Also, questions were presented individually on the screen. One would assume that older people would use the computer more difficult to use, but in this study elderly patients also preferred the touch-screen. All in all, the computer version produced better results, was easier to use, and preferred by the patients.

In another setting, Denny et al. [2008] found that usage of hand-held touchscreen devices for data collection was feasible in schools. Students had to fill out questions on touchscreen hand-held devices (Figure 4.1) and laptops. The largest part of the students had no preference between the two data collection modes. However, the ones that had one, strongly preferred the hand-held devices. The reason was ease of use and larger perceived privacy.

Dupont et al. [2009] came to a similar conclusion in a medical setting, where patients used a touch-screen computer to fill out questions about their quality of life. The questions that were asked ranged from family matters to questions about the partner and sexuality. Interestingly, questions about very private matters or other sensitive information were answered more openly by participants of both studies. In other areas, agreement between the paper and electronic

survey was almost perfect. Dupont et al. [2009, Page 95] suggested that patients may have greater trust in the security of their data if it were collected digitally. Another factor affecting this thinking may have been the presentation of the questionnaire. There was only one question on the screen at a time and users had the option to select their answer, after which the next one was displayed. With this approach, it is difficult for other people to see the previous content of the screen.

4.3.2 Touchscreen Text Input

Text input on smartphones is not as precise as on personal computers with keyboards. For example, Klemmerer [2011] found that text entry on a smartphone is prone to errors. Participants typically produce errors with capitalization and misspelled words. This can only be corrected to some extent by autocorrect features. The reasons are the small screen sizes and the fact that extended text entry has higher cognitive load than simple touching targets on the screen.

The situation on tablets is somewhat the same. Although there is more screen estate than on their smaller counterparts, effectiveness on touch-screen tablets with virtual keyboards is lower than on physical ones. Findlater, Wobbrock, and Wigdor [2011] argue that since there is no tactile feedback, users must constantly keep visual contact with the device, making typing slower. Errors are produced because of unintended touches with the keyboard as users are accustomed to resting their fingers or palms on the typing surface. All this leads to the conclusion that prolonged typing on mobile devices should be avoided when designing questionnaires and surveys for those devices.

4.3.3 Media

Mobile devices are widely used for digital media consumption. Nguyen and Chaparro, 2012 conducted an online survey on how students use tablets, in this case the Apple iPad, compared to non-students in their daily lives. They found that the former use the device for games, social networks, audio and video, while the latter use it more for reading magazines and eBooks. The study suggests that people who use such devices are familiar with digital media, and also use it the majority of the time spend with this kind of devices. Furthermore, there is published work about the usage and possible benefits of digital media in self-administered interviews [Watson et al., 2001]. While, again, such trends cannot be generalized, it should be save to assume that media can also be used to better questionnaires and interviews performed with mobile devices, when used in an appropriate context.

An interesting approach that becomes possible with touch-screen devices is the possibility of image elicitation. Here, a user or interviewee selects or marks certain parts of an image which is displayed on the screen. The data can be saved in a cartesian coordinate space for later analysis, making this a new question type.

Another possibility would be questions where participants are shown a video or hearing an audio recording. Afterwards, they can answer questions about it or talk about their impressions. This is particularly interesting because it can be embedded in the questionnaire, and the user is not taken out of context by, for example, switching to another application to play the media. The potential benefit of such approaches, again, depends on the context and the questionnaires or interviews, and further research in this regard has to be conducted.

4.4 Surveys with Mobile Devices

Two instances were found in the literature that used mobile devices for surveys. The first is from Klemmerer [2011], comparing surveys on mobile devices to the web and paper surveys by building a prototype application for the Android platform. Participants filled out surveys in the three different modes, and the results were compared to each other. The preferred mode was the web survey, followed after mobile and then paper. However, the mobile method was perceived as the most fun to use. Furthermore, participants needed more time for the completion of the mobile survey compared to others, and this was, in part, attributed to the prototypal character of the application with long loading times and several errors, as well as the small screen size.

Similar results were found by de Bruijne Arnaud and Wijnant [2013], also comparing the mode effect of surveys. A questionnaire was developed for mobile devices, with unique design considerations to account for screen size and the touch screen interface. While the results of surveys performed with the mobile devices were comparable to the ones from web surveys, the completion time was found to be longer than on their larger counterparts.

Chapter 5

Comparison of Interview and Questionnaire Applications on Mobile Devices

“There is just no comparison between having a dinner date with a man and staying home playing canasta with the girls.”

[Marilyn Monroe, American actress, 1926–1962.]

In this chapter, several applications for interviews and questionnaires are described, evaluated, and compared. Only the android and iOS ecosystems are used for evaluation because those two are the mobile operating systems with the widest adoption. Web based tools are only presented if they offer a mobile companion app, or if they are part of an ecosystem where web interfaces and mobile clients work together.

Because the field of interview and questionnaire applications is somewhat broad, software with many different focuses is presented. While they may not be comparable directly to each other, there are large areas where functionalities and features intersect with each other. Those should be illustrated, and certainly the usability and design of the applications are examined.

In section 5.1, the approach to the evaluation and comparison is described. In section 5.2 all the evaluated applications are listed. Descriptions, experiences, and classifications are given. Section 5.3 will provide an overview and stack the different applications against each other. Through this discussion, relevant features for mobile applications that deal with interviews and questionnaires are identified.

5.1 Approach to Evaluation and Comparison

The first step to comparing applications that deal with surveys, questionnaires, interviews, and polls is finding popular and good applications in those fields. For this purpose, the Apple AppStore and the Google PlayStore was searched with the keywords survey, poll, interview, questionnaire, question, and CAPI. The search engines of those stores offer metadata and ranking algorithms. By using this approach, the applications which seemed to have the best features, design and usability were selected for further comparison. The hardware for testing was an Apple iPad 3 for iOS and the Motorola Xoom for Android.

5.2 Applications

Since the breadth of the field, there are a large number of applications with a number of different focuses. The largest areas were applications for filling out forms, performing surveys, filling out short surveys for monetary incentives, and applications for interviews supporting audio and notes. Over 50 applications in those various categories were found, and the top eleven are described.

All listed applications were installed and evaluated, given that there was a free version or the app had a demo mode or questionnaire. The focus of the evaluation was with the usability, the provided features and also if the design fits the platform.

5.2.1 mQuest Surveys

When using mQuest survey, no demo account was needed because a sample survey was downloaded on the device by selecting it in the main menu. It is an universal application, running on iPhones as well as iPads and Android [*mQuest Surveys 2013*].

Surveys are created with a Windows application, which could not be tested because the link after the registration was not working; therefore the demo was evaluated. Several question types like multiple choice, checkboxes, matrix questions, text entry, ordered question selection, questions with images, video, audio, and numerical questions with consistency checks are supported. However, the demo application did not include multimedia in questions.

A questionnaire is started by selection the play icon from a list in the main menu, which is usually reserved for detailed information in iOS applications. The whole demo questionnaire could be completed. After answering the last question, the application immediately returns to the home menu. When answering questions, the next question is automatically displayed if possible. For example, if the question type is single choice, the next screen appears automatically, but this is not correct for text questions. Also, the matrix questions use checkboxes, which can lead to not seeing the answer options for longer questions. Users have to select an answer, or an error message is displayed. However, before showing this message, the whole screen transitions as if the next question will be displayed. This behavior is certainly confusing to a user. Also, selections cannot be deselected, only changed.

Numerical questions show the whole keyboard, and although letters cannot be entered, they can be tapped without anything happening. An implementation only showing numbers would have been better. Numerical values can be limited to particular ranges, resulting in an error if a wrong number is entered.

The performed surveys cannot be shown individually, only the total number of performed surveys is displayed. The whole dataset can be deleted, but not through swiping or an edit button, which is usually the case in iOS applications. This option is part of a menu which is used throughout the application. While the icon of this menu is always the same, the displayed options differ between screens, which is surely confusing for users.

The data can be synced with the web service, and no other option for export on the device is present. All in all, the application does not follow typical iOS patterns for designing user interfaces, but works good when answering questions.

5.2.2 Loop Survey Maker

Loop Survey maker offers the possibility to create surveys directly on the device [*Loop Survey Maker* 2013]. However, to start the application, creating an account is necessary. This is not recommended if the app would not strictly need an online connection and account system, but as it turns out, the application uses a web API to distribute surveys and receive answers. The focus of the app is to create short surveys for customer feedback, and this part would have been possible with an offline version.

When creating a survey, the user can select between predefined categories called restaurant, hotel, product, service, education, or create a custom category, which was done for evaluation. Once created, questions can be added with the options stars, text, Yes/No, numerical, empty (text entry), frequency (multiple choice with labels first time, weekly, monthly, yearly), smileys, comments, and an email field. Some of those can be seen in figure 5.1. An error was discovered when adding questions. Every type was displayed two times, once in english, and once in the language that is set for the device, in this case German.

When selecting the numerical question type, the numbers could not be changed. Only single choice selection of numbers between 1 and 10 is possible. Contrary to this, the text labels could be changed. The number of them could be increased or decreased, and the text of the labels was editable. This means that the blank question type is just a text single choice type with empty labels.

The questions can be edited, deleted and reordered (figure 5.1). The number of questions is limited to seven in the free version. After saving the survey, it can be filled out on the device. From the main menu, the survey then can be shared through the internet, Facebook, Twitter, Mail or SMS. The results of such shared surveys are immediately visible inside the iPad application. In addition, surveys can be conducted on the device. When doing so, the survey can be locked and only unlocked with a key. In conjunction with the kiosk mode of the iPad, participants are only allowed to fill out surveys and not see the rest of the operating system or the application. Also, a summary of all survey answers can be viewed with analytics for the different questions and answers. Survey data can also be exported through email in the Comma Separated Values (CSV) file format [*The Comma Separated Value (CSV) File Format* 2013].

5.2.3 TabSurvey

TabSurvey is divided into two parts, an online dashboard for creating surveys and analyzing data, and an iPad application where the surveys are performed [*TabSurvey* 2013].

Question types that are supported are smileys, loyalty (1-5 numerical selection), text, email, numerical and some variations of multiple choice questions. There are also options for de-selecting other options after the last selection. Where possible, a preference can be set that automatically moves to the next question upon answering. For every survey, a username and password is generated. Those can be used on up to ten devices. In addition to the iPad, surveys can also be performed in a web browser. For this purpose, a link is created through the dashboard which can be sent to participants per email. Report data could be exported in both CSV and Excel file formats. Also, the whole questionnaire could be exported as plain text, making it possible to share it with other TabSurvey users who can import it. Further more, questionnaires can be written with a plain text editor. The responses to surveys can be viewed in the dashboard and analyzed with the help of diagrams.

Figure 5.1: Loop Survey Maker - Editing a questionnaire: Rearranging and deleting different types of questions.

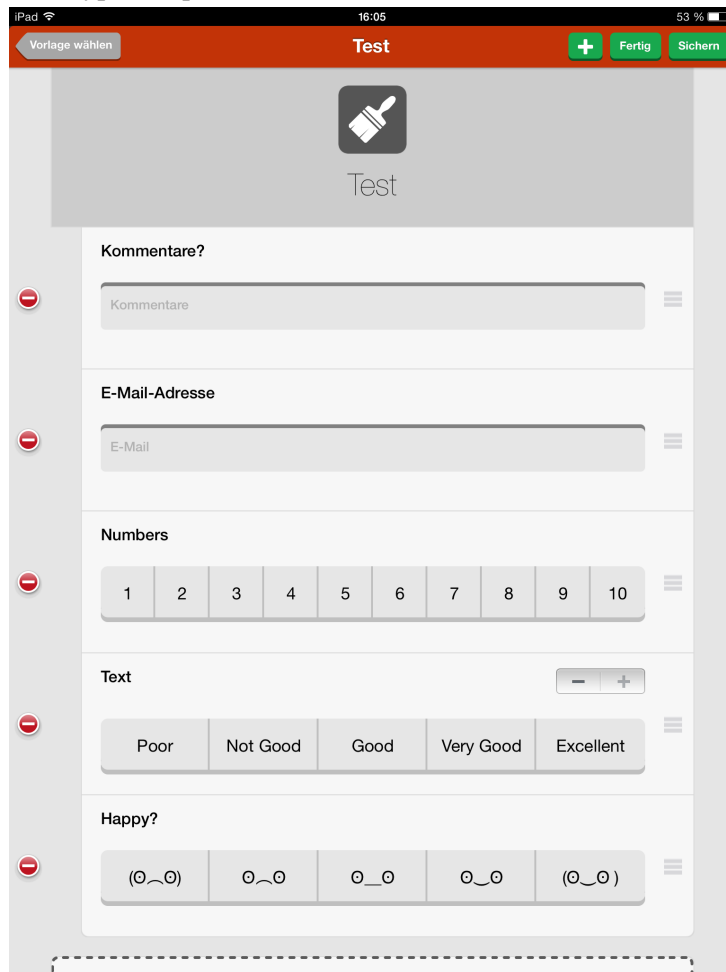


Figure 5.2: TabSurvey - Full screen with navigation and status. Single choice, multiple selection question.

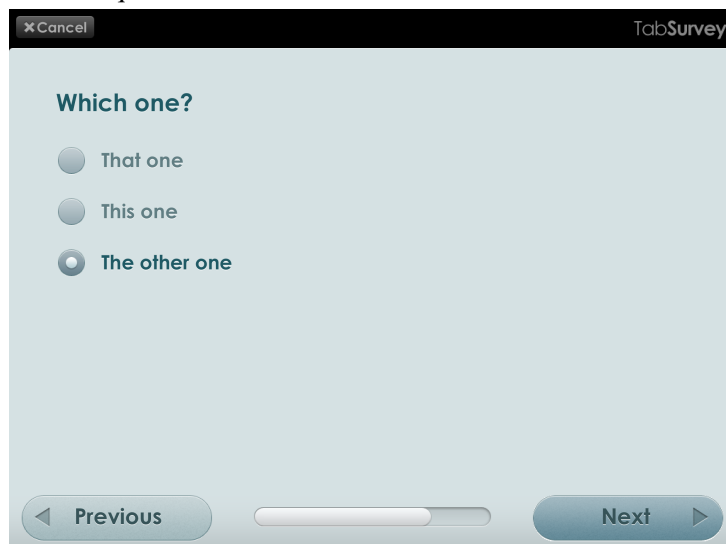
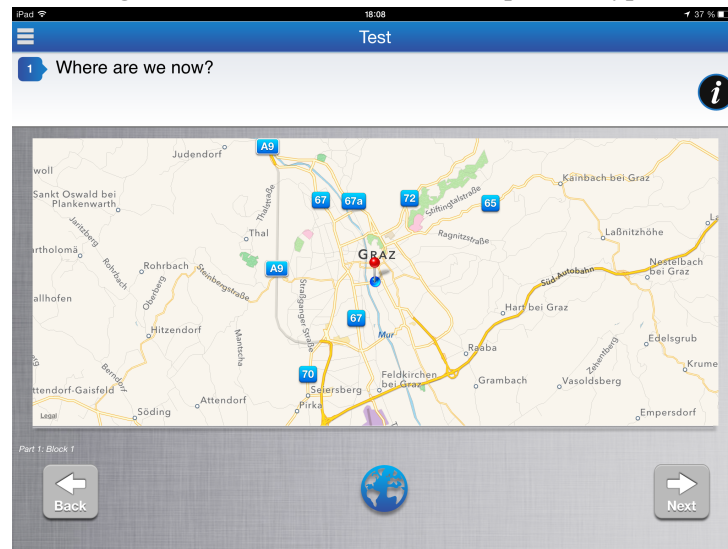


Figure 5.3: DataField - Location question type.

The interface is full screen, only displaying one question at a time. On the bottom, there are large next and back buttons as seen in figure 5.2. Questions where answering moves automatically to the next question disable those buttons. At the end of the survey, results are uploaded to the web service. A survey can be aborted at any time by pressing the cancel button in the upper left corner of the interface. One problem while trying out test surveys was, that a survey cannot be switched for another. For this, the app has to be restarted. Also, for the smiley questions the automatic showing of the next question upon answering did not work when set as an option in the dashboard. The email field is only accepting valid email addresses. Further more, only full numerical numbers can be entered, no decimals are possible.

The great advantage of this application is the validation of email and that there is only a numerical field when entering numbers, making wrong entries impossible. Also, ensuring a complete dataset, participants can only go on to the next question if they provide a valid answer.

5.2.4 DataField

DataField is available as an universal application on iOS [DataField 2013]. Surveys and forms are created through an online dashboard. The most interesting feature of this application is support for images captured with the mobile device, location data through GPS and signatures by drawing directly on the device as seen in figure 5.3. These come in the form of a unique question type which can be set in the dashboard when creating a survey. GPS can also be saved in the background by hiding the question. Other supported questions include multiple choice, single choice, date, time Yes/No, gender, likert, scale, number, text, email, and smiley. For every question, it can be set if it is mandatory. Also, instructions with a link to a youtube videos or an embedded picture can be added to questions. Also, conditional branching for questions is supported.

After the survey is created, a mobile campaign can be started. Surveys can be either public or private protected by a password. The starting time and duration of the campaign can be set in the online dashboard. A notification is sent to participating users having the app installed.

On the mobile side, participants need to enter credentials for access to the surveys. Those

credentials were created in the online dashboard. After this, a list of available surveys is presented. Tapping an entry starts the survey. Questions are in full screen, and there are large next and back buttons at the bottom of the screen. After the last question, all questions are displayed in a list for verification. From this screen, any question can be edited. However, if a user might want to edit only the first question, all other questions are displayed again. On the last screen, the user can tap a button in the upper right corner of the screen, which makes the data available in the dashboard for further analysis. From here on, the results could be exported through CSV or uploaded to Box, which is a cloud service for files [Box 2013].

Overall, the application works well and has unique questions that are only possible with mobile devices, but the design of the user interface is far from the quality of competitors.

5.2.5 Survey Master

With Survey Master, surveys can be created and performed on the device [Survey Master 2013]. By switching into a "Edit Mode" they can be created. Every time this is done, a message explaining the mode which needs to be dismissed is displayed. This message should not be displayed more than one time. What differs from the other applications is that surveys can also be exported, as well as imported through files. The supported questions are multiple choice with checkboxes, single choice with radio buttons (also with predefined labels), free text, mail, phone number and address collector, which is a form with multiple text fields. When adding a question, a small preview is displayed. Also, the question design can be customized from the menu by setting a background image from the library or by taking a photo. Furthermore, solid colors can be set.

After creating the survey and leaving the "Edit Mode", it can be selected from a list. Details like creation date, total questions, number of people who took the survey, number of completions, the last survey, and the date of the first survey are displayed.

After selecting a questionnaire, the application switches to a full screen experience. Similar to previous applications, there are next and previous button at the bottom of the screen. At any time, the survey can be aborted by pressing the cancel button in the top right corner. Also, completion messages can be displayed.

Results can either be exported through email or saved to local storage in a CSV format. The same is possible for the survey itself, making sharing between people and devices easy. Overall, the app works well, but the user interface gets somewhat in the way with unusual design decisions. Surveys cannot be changed once data was collected, which has to be deleted before making changes. The UI is heavily skeumorphic; mimicking real life objects like sheets of paper and college blocks. When mixed with digital only UI elements, this analogy to real world objects breaks down and can provide a suboptimal user experience.

5.2.6 SF-36 Survey

SF-36 Survey is specifically built to measure subjectively estimates of the quality of life of patients [SF-36 Survey 2013]. With the extensive research on touch screen questionnaires and surveys done in the medical field, this application could be of interest because of the know how that went into its crafting.

When starting the application, a new test can be started (2 times in the free version). The purpose of the first screen is to collect metadata for the patient and the investigator. Also, a

Figure 5.4: SF-36 Survey - Full screen questionnaire.

IPad 13:12 22 %
SF-36 Survey Seite 4 von 28

3. Im folgenden sind einige Tätigkeiten beschrieben, die Sie vielleicht an einem normalen Tag ausüben. Sind Sie durch Ihren derzeitigen Gesundheitszustand bei diesen Tätigkeiten eingeschränkt? Wenn ja, wie stark?

a) *anstrengende Tätigkeiten, z. B. schnell laufen, schwere Gegenstände heben, anstrengenden Sport treiben*

Ja, stark Ja, etwas Nein, gar nicht

b) *mittelschwere Tätigkeiten, z. B. einen Tisch verschieben, staubsaugen, kegeln, Golf spielen*

Ja, stark Ja, etwas Nein, gar nicht

c) *Einkaufstaschen heben und tragen*

Ja, stark Ja, etwas Nein, gar nicht

Zurück Weiter

numeric pin can be set in this form with which access to functions and menus outside of the test is limited, making it impossible for patients to leave the test.

When starting the interview, a welcome screen is displayed. By pressing the next button on the lower right corner, the survey begins. Figure 5.4 shows the User Interface (UI). Again, like it is described in the literature and implemented in various other applications, a large next and back buttons are used for navigation. On the top right corner the current question number is displayed, as well as the number of remaining questions. Interestingly, this textual view alternates with a visual progress bar every couple of seconds. The questions have to be answered, or an error message is displayed. An alternative to this approach would have been greying out the next button and only activating it when a user answers the question.

The question types included are single choice with vertically aligned checkboxes and scale questions with text labels. Deselection is not possible, and sub questions are supported. At the end of the survey, the next button changes its label to "Done" which takes the user to a field where unlocking the menu with the previously provided pin is possible. Once a survey was started, it could *not* be aborted.

The performed surveys can be opened for data analysis on the device. Graphs are created which compare the patient's performance to mean values of a greater population. The report can be send per email in the form of a PDF file and can also be password protected. This representation is supplemented with the data in the CSV file format.

While this application offers a customized report, which is augmented with data outside of the data collected through the survey, importing other questionnaires is not possible, limiting the use to this health quality survey. For broader acceptance, such applications should allow the import or creation of any survey. Therefore, they have to support a wide array of possible question types and import and export functionality.

5.2.7 TouchField

TouchField is available both for the iOS and Android operating system [*Opiniometer Touchfield* 2013]. It works in conjunction with a web interface called Opiniometer which serves as a hub for creating surveys, organizing users, and analyzing data. This web interface, or dashboard, includes a large set of features, even letting administrators create previews of tablet and smartphone surveys and try them out in the web browser. All surveys can also be printed or published on the web.

Surveys support branching and piping with logical criteria. Images, video, and audio can be added to questions. Responses can be selected to be mandatory, giving administrators or survey creator granular control over the flow of the whole questionnaire. However, surveys have to be created for different formats and devices. Those are web, paper, smartphone, and tablet, each offering different capabilities. A sample survey for tablet devices has been created for the purpose of this thesis.

The supported question types are extensive, and they include multiple choice, both vertical and horizontal, choose all that apply, open ended questions, and a wide range of matrix questions. In addition to those options, question ordering can be randomized, automatic proceeding to the next question upon answering added, and setting the question as required can be activated.

The demo survey was tested on an Android device. After adding the credentials, the device has to be registered by providing a name, email and location. In the next step, the demo survey is automatically downloaded and started. Questions are presented in full screen, and there is a next and back button at the bottom. In addition, the current and remaining questions are displayed, both as numbers and as a progress bar.

One problem when trying to perform a survey was, that after not interacting with the device for about half a minute, a pop-up appeared saying the survey timed out and it was restarted from the beginning immediately without user interaction or the possibility to abort this process. This is very problematic in a scenario where participants think about their answers or want to discuss them. Furthermore, the interface seems not optimized well for different screen sizes. The answer options are in a picker menu (similar to drop-down), when the whole screen could have been used. Ideally, this question type should not be used. There is also a question type where users can add comments through text, audio, image or video, but questions with media embedded could not be tested. At the end of the survey, the data is uploaded automatically. It is then available in the web interface for further analysis. Results can also be exported using various formats, including CSV and Excel files.

5.2.8 iSurvey

The application iSurvey works both on iOS and on Android [*iSurvey* 2013]. However, the application is called droidSurvey for the latter. Surveys are created on the web portal. When creating a survey, several branding options are available, and a preview of how the survey will

look on the devices is available. GPS location can be automatically tracked and uploaded upon completion.

This tool offers a larger selection of questions, including text, interval, email, information, numeric, image, single selection, date and time, multi selection, signature, scale, barcode, drop down and matrix questions for single selection, multiple selection, and scale questions. Also, instructions can be added for every question in addition to the question text.

When testing the demo survey on the device, in this case a tablet, the first thing that is evident is that it is not a native application, and it is not targeted for tablets, but for smartphones. The UI is a blown up smartphone interface. Apart from that, the application performs well. Next and back buttons are at the top of the screen, and there is no option to abort the survey prematurely. Date questions are answered using a picker, making wrong entry impossible. Great features are the likert scales and intervals. A slider can be dragged from left to right and above it the current text label will appear. For this type, images are also possible. In the demo, the question uses smileys to indicate satisfaction or dissatisfaction. Email addresses have to be valid to proceed, and numerical questions are answered with a custom numerical pad interface, again making it impossible to enter wrong values. No decimal numbers are supported. At completion, the application returns to the main menu where the data can be uploaded. It can not be viewed on the tablet. Export options include CSV and SPSS. Surveys can be completed offline and uploaded at any time.

5.2.9 Surveyor

Surveyor is available for the iPhone, iPod Touch as well as the iPad [*Surveyor* 2013]. The ecosystem is again tied to a web interface for survey creation and result collection. However, surveys can be performed offline on the device. Skips and jumps are also possible for questions, as well as the option for mandatory answers.

Right on the device there is a demo survey with all the available question types. The custom design of the application is a great plus for user friendliness, as the questions and answer options are in full screen and they are good readable. There are arrows for question navigation in the top part of the UI, and the survey can be aborted at any time with a button on the bottom left part.

There is a fair amount of question selection. Multiple choice selections can be mutually exclusive, and answers can also be deselected, making it the only app in the list to support this feature. Text questions are supported. Numeric question show a keyboard with numbers and other characters, and there is no validation, making it possible to enter wrong data. Multiple choice questions are also offered with a text field that the user can fill out freely. A GPS question is supported, showing a map with the current location where users can mark points.

Overall the application works very well and has a sophisticated user interface, both at the client side when performing surveys, and at the server side where surveys are created, and the results are captured. These can also be exported in a CSV format.

5.2.10 WiseHuch

WiseHunch is only available for the iPad and is the first, and only, application in this list specifically built for interviews that also supports audio recording [*WiseHunch* 2013]. However, it

has a very narrow focus as its main use case is to conduct interviews about problem assumptions and solution assumptions used for product development. The audio recording and multiple choice questions are premium features and could not be evaluated; nonetheless those features are described in this section.

For both interview types, problem assumption as well as solution assumptions metadata questions can be added to find out about the demographic. Those contain questions about gender, age group, occupation, income, marital status, and three Yes/No questions with custom text. Those are fixed, and no further questions can be added in this category.

In the problem assumption interview, the first step after the demographic data collection is the problem context. Essentially, there are four text fields where questions can be entered that are asked during the interview. After this, the problem rating section follows. Three identified problems from the previous question can be entered and rated with a scale and slider which can be set at the values not rated, no need to solve, nice to solve, and need to solve. Following this, the next questions are about discussing current workarounds. Finally, in the last section, the permission for follow up interviews is asked and name, email, and the phone number can be entered.

For the solution assumption interview, three problems can be filled out, essentially making them text questions. The purpose of the next section is to show a prototype described by a unique value proposition and three features in the form of text. Images can be added to this question. At the end, permission to follow up is asked like in the problem assumption interview.

During the interview process, notes can be entered by the interviewer, but only for some sections. The audio recording feature allows recordings of up to half an hour. Markers can be set during this process to find key passages later, which can be seen in figure 5.5. Those are numbered and can be moved, added, or deleted after interviews. The usefulness remains unclear as no label indicates why there is a marker, making it difficult to understand their meaning for anyone else than the interviewer.

The interface is completely custom and does not follow the Human Interface Guidelines (HIG), which are recommended for iOS applications. It tries to mimic maps and drawers, but goes so far with it, getting in the way of a fluent and fast navigation. After completing interviews, reports can be exported as PDFs. It was not possible to see if the audio and the associated markers could be exported using the free version. Furthermore, the description of the application also did not indicate such functionality.

5.2.11 FluidSurveys

With FluidSurveys [*FluidSurveys* 2013] surveys can be performed on all iOS and Android devices and they are created using a web interface. In addition to the ones created there, surveys can also be imported from the popular survey services SurveyMonkey and Zoomerang [*SurveyMonkey* 2013; *Zoomerang* 2013]. In addition to custom creations, several templates for surveys in the areas customer satisfaction, education, market research, and others are offered.

Among the supported features are jumps and conditions for questions and random question ordering. Question types are numerous and can be added through drag and drop. Supported are Yes/No questions, text, multiple choice with radio buttons, and checkboxes, with all of these also available in a grid view. The advanced questions were not available in the free version, but they are very diverse and use the unique properties of mobile devices well. For example, there are questions supporting sliders, timers, date/time, rankings, JavaScript, file upload, signature,

Figure 5.5: WiseHunch - Record audio during interviews and add markers to it.

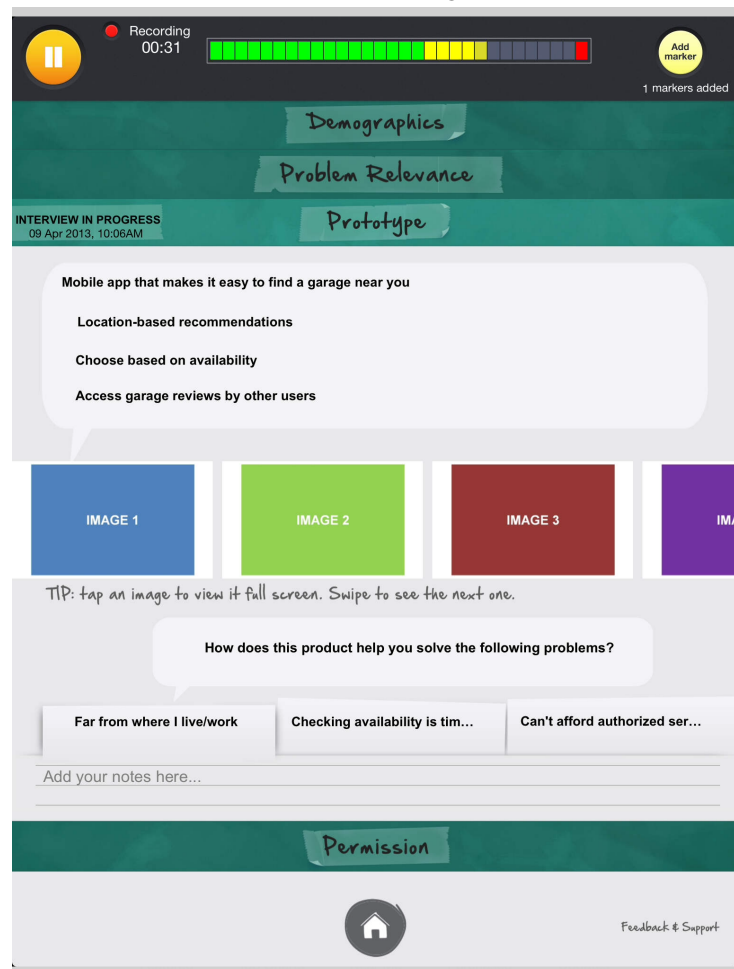
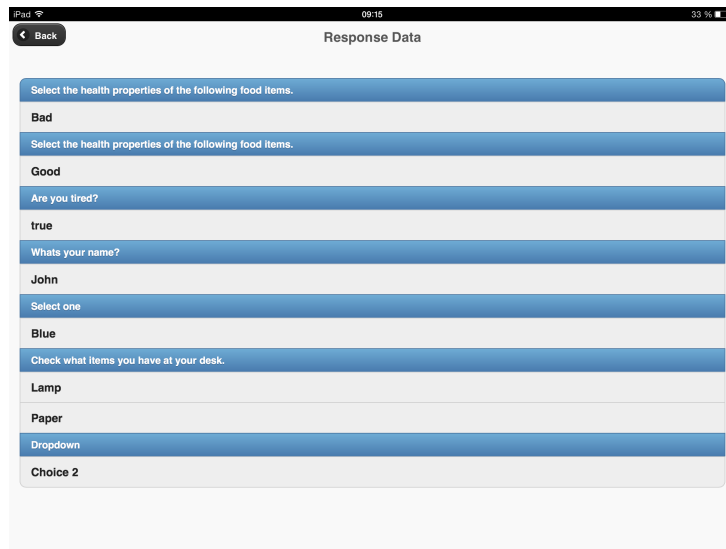


Figure 5.6: FluidSurveys - Survey results on the device, but without the other answer options.



pictures, and even payment questions. All together, there are 31 question types listed on the homepage. After the survey is finished it can be published with a link or through several social media channels like Twitter, Pinterest, Tumblr, and others. In addition, it is now available on the mobile devices.

When opening the FluidSurveys application on the iPad, a login screen has to be filled with the email and password that was used for registration. After logging in and syncing with the service, the created surveys are ready to be started. After selecting a survey, responses can be viewed, uploaded, or a new response can be added. After starting the survey, all the questions are displayed in a list. There is not a full screen mode with one question at a time. The survey can be locked with a password, disabling and hiding all the navigational interface elements. It can be unlocked by touching all four corners of the device.

The design of the questions is very clean. Just black text on a white background, making it easy to read. This is the standard theme, and others can be selected in the online interface. The survey works in landscape as well as in portrait orientations. When scrolling to the end of a survey, the response can be uploaded after which the survey begins again.

As already mentioned before, responses for the survey can be viewed on the device. However, not only those collected offline can be viewed, but also all of the responses that are stored online, which can be seen in figure 5.6. This worked well for the first time, but resulted in a stuck UI when trying to access the data at a later time. Also, sometimes surveys were not displayed, and only syncing data would make them appear. While the UI does not use native interface elements, it works well and has a large feature set with many questions. Results can be exported with file support for SPSS, CSV, Word, and PDF. Reports with a wide variety of charts can also be generated online. The FluidSurveys application is one of the most extensive survey application evaluated. It offers numerous questions and a good design for both mobile and web.

Table 5.1: Comparison of survey applications by platform and features.

	mQuest Surveys	Loop Survey Maker	TabSurvey	DataField	SurveyMaster	SF-36 Survey	TouchField	iSurvey	Surveyor	WiseHunch	FluidSurveys
iOS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Android	✓			✓			✓	✓			✓
Create Survey on Device		✓			✓						
Create Survey in Web Interface			✓	✓			✓	✓	✓		✓
Create Survey on Windows	✓										
Full Screen Questions	✓		✓	✓	✓		✓	✓	✓		
Offline Support	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Randomized Responses				✓			✓	✓			✓
Required Response	✓		✓	✓			✓	✓	✓		✓
Audio Recording										✓	
Branch/Jumps Logic	✓		✓	✓			✓	✓	✓		✓
Questionnaire Import/Export			✓		✓						
Lock Screen		✓				✓					✓

5.2.12 Qualtrics Surveys

A demo account was requested for the Qualtrics Surveys mobile application, but not received in time [QualtricsSurveys 2013]. Therefore, testing could not be done. The mobile applications are companion apps for performing surveys which were created through an online interface. The latter has numerous options for customization, theming, looping, logic criteria, multimedia, and question types. The site claims that here are over 100 question types available. Because of this extensive catalog of question types and other features the application is listed here, even though it is not part of the comparison.

5.3 Comparison and Results

To get a comprehensive overview of the applications, the comparison has been divided into several groups. In the first section, shows what the applications and ecosystems have in common when creating questionnaires and what features they offer. The second section shows what questions types are available. Most services offer the sharing of surveys not only through their mobile clients, but also on the web. The last section shows the offered reporting, import, and export features.

5.3.1 Creating Surveys and Questionnaires

Most surveys can only be created in a web interface (6/11).

Form the eleven examined applications, only two offered the ability to create surveys on the device (Table 5.1). Even if such a function is offered, the questions and question types were very limited in comparison to the web interface counterparts. This may have to do with the fact that content creation is still considered faster on PCs than on mobile devices. It is possible to create more complex questionnaires on a larger screen and with a keyboard and mouse, at least for now. Therefore, there is a large vacuum that could be filled by applications on mobile devices that cater to that need.

All applications offer offline capabilities (11/11).

This was the most obvious feature as it is supported by *all* the evaluated applications (Table 5.1). Offline data collection seems to be a must have feature. Even applications that are clearly not native, but hybrid approaches offer this functionality. This is the number one feature on the list when considering applications for surveys and interviews.

The majority of applications offer skip/jump/filter logic (7/11).

This seems to be a very important feature as it is offered by seven applications (Table 5.1). The literature review showed, that computer assisted approaches can offer more value to interviews, self administered interviews, and questionnaires. To do this, technology has to be used in a way where classic approaches have problems. This can be especially true for skip/jump/filter logic, which is clearly when using a computer assisted method. Data checking and validation are also very relevant in this context.

Most are full screen with one question at a time. (7/11)

Of the evaluated application, seven offered a full screen experience with one question at a time (Table 5.1). Those have the interface elements for question navigation in common. Nearly all the questionnaires could be navigated with a next and a back button on the bottom of the screen. Only one application placed the buttons on the top. Importantly, the size of the buttons was fairly large, which is also in accordance with the findings in the literature. Furthermore, the option to automatically proceed to the next question upon answering was provided in two products.

There were application specifically for kiosk use. (3/11)

Two features are essential for kiosk modes. The first is locking the screen so that a user cannot accidentally leave the application or navigate administrative menus. This was achieved through passwords and pin code. The second feature is the questionnaire looping. When a timeout is reached without user input, the questionnaire is started again. However, this timeout was too short in one of the applications (Table 5.1).

Table 5.2: Comparison of question types.

	mQuest Surveys	Loop Survey Maker	TabSurvey	DataField	SurveyMaster	SF-36 Survey	TouchField	iSurvey	Surveyor	WiseHunch	FluidSurveys
Single Choice	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Multiple Choice	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓
Likert/Scale	✓	✓		✓				✓		✓	✓
Matrix	✓						✓	✓			✓
Numeric	✓		✓	✓				✓	✓		✓
Date/Time				✓				✓			✓
Text	✓	✓	✓	✓	✓			✓	✓	✓	✓
Email		✓	✓	✓				✓			✓
Order	✓										✓
Location				✓			✓	✓	✓		
Multimedia				✓			✓	✓		✓	✓
Drawing/Signing				✓				✓			✓

Only two applications offered importing/exporting of questionnaires (2/11).

Only two application supported a file format which defined a questionnaire (Table 5.1). Interestingly, one of it was also one of the apps that offered the functionality to build surveys on the device. With this approach, it would be possible to share surveys between different software products, if they would support the file format. With such an approach, researchers, interviewers, and survey makers can freely share their created surveys between different applications if a file format was widely enough adopted.

5.3.2 Question Types

Most applications offer single/multiple choice questions (10/11)?

These two seem to be one the most commonly used question types, and 4 applications also offered this kind in a matrix form (Table 5.2).

Multimedia can be embedded in questions and questionnaires (5/11).

Multimedia through images, pictures and video could be embedded in five applications (Table 5.2). Also, one application offered the functionality to answer with images saved on the device or from the camera. Some applications offered the option of theming and setting background images, but this was not quantified. This feature, along with location, has a high potential for innovation.

Table 5.3: Comparison of import and export capabilities.

	mQuest Surveys	Loop Survey Maker	TabSurvey	DataField	SurveyMaster	SF-36 Survey	TouchField	iSurvey	Surveyor	WiseHunch	FluidSurveys
Reports on Device		✓		✓	✓	✓				✓	✓
Reports in Web Interface	✓		✓	✓			✓	✓			✓
Graphs on Device		✓				✓					
Graphs in Web Interface	✓		✓					✓			✓
Export on Device		✓			✓	✓					
Export in Web Interface			✓	✓			✓	✓	✓	✓	✓
File Format CSV		✓	✓	✓	✓	✓	✓	✓	✓		✓
File Format SPSS								✓			✓
File Format Excel			✓				✓				✓
File Format PDF						✓				✓	✓

Location tracking was offered in the applications (4/11).

Of the evaluated applications, four offered questions with some kind of location integration (Table 5.2). For example, participants could add their location on the map. Furthermore, three applications offered the functionality to track the location in the background. Mobile form clients offered fields for signing. Those three instances where this on the screen is possible are only used for signing of forms. No app had the functionality to let users draw as an answer.

5.3.3 Import/Export of Data

A large number of applications offered reports directly on the device (6/11).

Of the examined applications, 6 showed survey results on the device (Table 5.3). However, the trend was to only show the selected answer, not all of the possible answers together with the selection. Of the examined tools, only two offered complex graphing on the device itself. Most only offered graphs in a web interface. Again, like with the questionnaire and question creation, it may be desirable to offer this feature directly on mobile devices.

The most popular data export format is CSV (9/11).

The format CSV is the most commonly offered file system to export survey data, followed by Excel and PDF (Table 5.3). SPSS, while often used in research, was only offered by two services. Giving users more choice for data export is desirable.

Chapter 6

TCAPI: A New Tool for Interviews and Surveys

“I believe in innovation and that the way you get innovation is you fund research and you learn the basic facts.”

[Bill Gates]

This chapter describes the creation and evaluation process for the Tablet Computer Assisted Personal Interviewing (TCAPI) prototype application. It is built to offer useful features for qualitative and quantitative methods. The project, which was under the scientific administration of Prof. Manfred Prisching, was conceived and organized by Rafael Schögler and Martin Griesbacher at the Centre of Social Research at the Karl-Franzens-University of Graz. The software development was done in cooperation with Evolaris Next Level GmbH. Together with their initial project plan, findings in the literature, and the application comparison a prototype was developed. This chapter is devoted to the development process and evaluation of the prototypes potential. Development was done in cooperation with Evolaris Next Level GmbH. While the initial focus of the prototype is to be a tool to supplement interviewers, quantitative surveys and questionnaires should also be possible for this first iteration. Current trends in mobile design and the special capabilities of mobile devices were taken into consideration over the whole development process.

The chapter is divided into five sections. In section 6.1, requirements for the prototype are presented. Following this, possible features are described and discussed in section 6.2. Those were rated for their complexities and priorities in section 6.3. Section 6.4 describes the chosen platform, ecosystem, development process, and the final prototype. Finally, the potential of the prototype is evaluated in section 6.5.

6.1 Requirements

6.1.1 Requirement 1: Useful for both qualitative and quantitative methods.

The initial project plan envisioned TCAPI as a tool to assist interviewers and conduct partly structured interviews. One integral part of this strategy is to foster mixed method approaches

by the use of new technologies to create new opportunities. This makes it crucial to envision a system that can also be used for quantitative methods. Hence, making it a requirement to support both methods and their combination.

6.1.2 Requirement 2: Support a wide range of question types.

The more question types an application or tool offers, the stronger are the benefits for standardized methods as more complex questionnaire designs are possible. Such a tool gets more useful if researchers can choose between a wide variety of possibilities. Mobile devices have some capabilities which are stronger than in other computer products, like their weight and portability, and some that are weaker, like text input and screen size. Those need to be taken into consideration when implementing different question types.

6.1.3 Requirement 3: Perform interviews and surveys offline.

Even though mobile devices have these excellent capabilities for wireless communication, there are times when a user is not able to connect to the internet. For that purpose, the application should be able to operate completely without an internet connection. Communication capabilities should only be seen as a natural extension, not as the only means of performing interviews and surveys, or collecting and analyzing data. This requirement also incorporates to keep as much functionality as possible on the device. For a tool to be useful anywhere and anytime, users should be able to do as much of the offered functions on the device itself.

6.1.4 Requirement 4: Use special capabilities that only mobile devices support.

Fully using the capabilities that mobile devices is an essential requirement for this work, and therefore also a strong requirement for the development process. A native approach must be chosen for development, because of access to the hardware Application Programming Interface (API) that is provided by the OS. Most of those are, for now, only available to native applications. The downside to this approach is that a native application cannot be run on other platforms, which limits its audience. However, there are hybrid approaches that can be used to create applications for multiple operating systems that still have access to hardware capabilities, but those are not the focus of this work.

6.1.5 Requirement 5: Focus on ease of use.

Focusing on ease of use means that a sophisticated User EXperience (UX) has the same priority as the features. Adding the latter should never be at the expense of the former. However, sophistication does not mean complex in the context of UX, but rather it means making complex systems easy to use, and there are many ways to achieve this. For example, the first law of usability as defined by Krug [2006] is "Don't make me think!", which calls for software products that are simple to use and instantaneously understandable. While he focuses on web pages in his book, the same principle is applicable to any UI. More specifically, he defines his first principle as, "It means that as far as humanly possible, when I look at a Web page it should be

self-evident. Obvious. Self-explanatory. I should be able to 'get it'-what it is and how to use it-without expending any effort thinking about it." Furthermore, [Krug, 2006, Page 34] positively emphasizes the use of conventions as they are developed over time, and universally understood by users. Staying close to tested conventions makes using software products easier to users, opposed to reinventing the wheel every time. Mobile operating systems have developed their own conventions, which brings us to the next requirement.

6.1.6 Requirement 6: Keep close to the paradigms of the operating system.

Mobile operating systems have unique ways for navigation between content and interaction with certain UI elements. Those conventions are unique to the respective OS. When developing an application for it, they should be used in most instances. The reason being, that users are used to them if they are familiar with the platform. Hence, even if they are new to a platform, those interface paradigms are a great guideline for crafting user interfaces, and should therefore not be ignored. The software vendors of both Android and iOS operating systems have created guidelines for their platforms. For Apples iOS, there are the HIG [*Human Interface Guidelines* 2013]. Google offers a similar guideline for their operating system on the developer homepage [*Android Design* 2013].

6.2 Features

The TC-API project with user stories were developed at the Centre of Social Research at the Karl-Franzens-University Graz under the scientific administration of Univ.-Prof. Mag. Dr. Manfred Prisching, by Rafael Schögler Mag.phil. Bakk. MA, and Martin Griesbacher Bakk. MA. They were responsible for the concept and were evaluating developed versions of the application. The user stories created by them can be found in appendix A. Together with those, the findings from the literature survey, and the comparison of the existing application in this field the specifics of the implementation were derived. Following from this, the complexity and priority of the features was established, providing a clear set of functionality that was implemented for the prototype application. This process is described in this section through detailed descriptions of every considered feature.

6.2.1 Import questionnaires

The central aspect of the whole application is the survey or questionnaire. This requires a way to describe such a questionnaire. At first, using EXtensible Markup Language (XML) for the document description was considered. XML is in a human readable form, making it easy to understand. However, creating a description of the whole document structure for questionnaires with sections, titles, comments, questions, as well as numerous question types, seemed unreasonable for the scope of this project, especially in contrast to all the other features that were also relevant. Even if a new data description is created, the need to create questionnaires would still be there.

Therefore, existing approaches needed to be considered. For this purpose, LimeSurvey, which is a popular open source tool for creating and distributing surveys through a web interface

[*LimeSurvey: Open Source Survey Creation and Distribution 2013*], was examined. Not only is it possible to create questionnaires with over thirty question types, they can also be exported in two file formats. One is queXML, which is also open source. It is a simple XML schema for designing questionnaires [*queXML: Open Source XML Questionnaire Schema 2013*]. The other is the survey structure file format (.lss) which is used by LimeSurvey internally. The difference between those two is that the survey structure format supports all of the lime survey features and can also be used to import surveys into LimeSurvey while queXML does not support as many features and question types. Nonetheless, queXML is much easier to understand and a comprehensive documentation is available, making it the preferred format for the prototype application.

Even with this approach, this is still a complex feature because the questionnaire schema needs to be validated, mapped to the applications internal database and rendered correctly on the screen. However, by choosing open source software, the ramp-up for development is much faster than developing own solutions and many additional features can be considered.

6.2.2 Create questionnaires on device

This is probably the most complex feature on the list because the database as well as the user interface for such a functionality must be very sophisticated. For every question type, a user interface for creation would be needed. This may seem trivial for simple questions like Yes/No or text answers, but it becomes complex if questions with sub questions are to be supported. Multidimensional matrix questions also have to be supported with such a feature. This was one of the reasons why queXML was chosen as the format for the questionnaires. With LimeSurvey, there already exists a popular open source tool for creating complex questionnaires and questions and it has the ability to export questionnaires with the queXML format.

6.2.3 Start, abort, continue survey or interview

After a questionnaire is imported, it should be possible to perform a survey or interview with the device. Not only should it be possible to start and end surveys and interviews, but also abort and continue them at any time. To incorporate this, the interviews that are saved in the internal database have to have a status. By this approach, they can be listed by this status. Open interviews at the top, and already closed interviews or surveys beneath it. With this approach, the interviewer can freely navigate the whole application as well as other ones, for example, to show participants a website. Continuing up the interview is possible at any time.

6.2.4 Questionnaires and results on device

Since one requirement of the application is to perform interviews and surveys offline, the data has to be saved locally. Building on this premise, the only thing left to do, is to design a user interface which lists interviews and the questionnaire. Also, it should be possible to view the questionnaire without starting an interview.

6.2.5 Many question types

Important for the standardized part of the application, is a wide variety of question types. The previous chapter showed that sophisticated applications offer the largest choice amongst different question types. The schema queXML also supports a rather large selection of question types, making it a great fit for a mobile application. However, the smaller screen size in comparison to web surveys has to be considered when implementing these questions, and the interface elements also need to be kept consistent with the overall interface of the selected operating system, as described in the requirements.

6.2.6 Record and play audio

In face-to-face interviews, conversations are often recorded for later transcription. This fact makes audio recording one of the most important features. The central thing to consider is that the conversation is clearly understandable. Also, the recording has to continue in the background, even if the screen of the device is locked, or the interviewer or participant switches to other applications. Another thing to consider is when audio or video is played inside the application that the recording remains of such a quality that the conversation is still understandable.

6.2.7 Export results in CSV

One of the major points of the concept, and also used by most applications and services that were compared to each other, is the export of data through CSV. This feature has a high priority and is also straightforward in its implementation. It offers users the ability to analyze their data independent of software solutions and operating systems.

6.2.8 Import and export of files

When considering how data is copied to and from mobile devices, their unique capabilities need to be considered. Clearly, as their name suggests, they are mobile and offer comprehensive communication capabilities through Wireless Local Area Network (WLAN) and mobile internet connections. Those capabilities are used for file exchange in the prototype application. Since only files need to be exchanged, existing cloud services can be used. Dropbox looks like a fitting candidate for the task, due to its popularity as described by Hunsinger and Corley [2012]. Also, it offers a developer facing API so that the service can be integrated into own applications. A more detailed description of those and why this service was chosen can be found in section 7.4. In addition to a cloud service for file exchanges through the internet, a tethered connection with a PC has to be offered in the case that there is no internet connection.

6.2.9 Question status

Catering to the requirement that the application needs to be usable as an assistive tool for open interviews as well as for standardized surveys, the answer status of questions and sections should be indicated visually. The reason being that, in weakly structured interview situations, interviewers tend to jump between questions and sections. Therefore, they need a visual indicator for answered and unanswered questions and sections. Since questionnaires can also be

divided into sections, those sections should also indicate if all of the contained questions were answered, or only a subset of them. For this purpose, the metaphor with read and unread statuses that are used in email applications was adapted accordingly. Unanswered questions and sections should be indicated through a circle, partly answered sections through a half circle, and fully answered sections and questions with no circle.

6.2.10 Full screen mode

This feature is relevant to the quantitative part of the application. It enables the application to be used for mixed method approaches. An interviewer hands over the device so that interviewees can answer questions on their own. No application could be found during the literature study that fulfills such a unique use case, making it a highly prioritized item on the list. Furthermore, this approach could offer methodological innovations.

The UI of the Full Screen Mode (FSM) displays one question at a time. Also, a next and back button is on the bottom of the screen, used to navigate between questions. In addition to navigation, the current question number and the number of all questions can be displayed to users, which is also a feature many applications from the comparison provided. In an interview situation, the cooperation between both parties is assumed, therefore, the need for required answers is not so high. If the questionnaire is self administered, this would be a useful feature, together with locking the application in the FSM. However, the latter features were only seen as a "nice to have" in the prototype application. Another feature of the full screen mode, is the navigation with gestures, which is a hallmark functionality of mobile devices. While they offer many multitouch gestures, a simple swipe left or right over the screen to navigate between questions is the most obvious implementation. Accompanying this, the screens should move in the right direction, giving more depth to the navigation gesture.

6.2.11 Location data

Also catering to the requirement to use technologies that are only possible with mobile devices, this feature is of central importance. While not in the initial proposal, this technology shows great potential in surveys and interviews, especially for researchers and marketers who work in the field. The position of the device can be tracked in the background for every performed interview or survey. Not only does this make obtaining the location for metadata very effortless, it also offers an additional dimension of data, supplementing the one collected through questions and interviews. Bringing those together could give new insights into answers and participants thoughts, as their environment could also be considered during interview situations. The complexity for implementing such a feature is low, with APIs that offer convenient access to the current position. In the comparison between application, it was shown that collecting location with GPS and questions that use maps already exist, indicating that there is demand for this feature in survey applications.

6.2.12 Multimedia in questions

Also using unique capabilities of mobile devices, embedded multimedia is an obvious addition to questionnaires and surveys on mobile devices. Questions augmented with images, video, and audio in questions or answers are an important addition to the application because it is only

possible on mobile devices. Furthermore, this is also documented by the wide use in already existing applications. Nevertheless, implementing such features is high in complexity, as the whole ecosystem has to support it. The standard queXML and its export through LimeSurvey do not currently support multimedia, making it necessary to implement new question types in LimeSurvey, as well as extend the queXML schema. Finally, the same thing has to be done on the client side. Having said that, even with the considerable effort to implement such a feature, it is one of those that offer the most potential for innovation. Those features were in the very first concept of the application.

6.2.13 Skip or hide Questions

This feature is also in the initial concept of the TCAPI application and is also part of any sufficiently complex tool for surveys as pointed out in the comparison of applications. Like the title suggests, logic criteria can be defined for questions, for example, only showing certain questions if the respondent is male or female, essentially making the questionnaire more dynamic. This feature is especially interesting for qualitative interviews, as the interview process is not only formed by the instinct of the interviewer, but can also be incorporated in the digital guideline which is used during the conversation. The complexity to bring this feature to an application is unfortunately also very high, as again the whole ecosystem consisting of questionnaire schema, creation software, and client application has to support it. While LimeSurvey offers such logic for the questionnaires that are build with it, it does not export it to the queXML schema, which in turn also does not support hiding of questions.

6.2.14 Tags or bookmarks in audio

Also in the concept was the ability to add tags to the audio recording whenever a jump from one question to another was performed. This should happen automatically while the interviewer navigates through the questionnaire. By this approach, transcribing interviews can become faster as users can easily switch to parts of the interview that are of interest to them. In the comparison, only one application for interviews emerged with a similar ability. It offered the possibility to add bookmarks manually to recordings. However, with the automatic approach, the interview flow is not interrupted by making the interviewer look at the device twice, once for selecting the next question and once for adding a bookmark. She or he can fully concentrate on the interviewee and the conversation.

6.2.15 Question filters

This feature lets interviewers filter questions by their answer status. In addition to the visual indication for the answer status of questions and sections a filter option should be provided. By selecting one of the options (answered, partly answered, not answered) from a menu, only the associated questions are shown. For example, an interviewer can only show the unanswered questions at the end of an interview. The current approach with pen-and-paper questionnaires or guidelines is to go through all the questions again and look what might have been missed. Of course, the benefit of such a feature only becomes apparent in longer interviews with a large number of questions or topics. The complexity of incorporating such a feature is of course depending on the chosen operating system and the APIs offered to the developer.

Table 6.1: Selected Features for Implementation

Feature	Priority	Complexity	Implemented	Reason
Import questionnaires	***	***	Yes	Essential for this type of application
Create questionnaires on device	*	***	No	Existing ecosystem was chosen
Start, abort, continue survey or interview	***	**	Yes	Essential for great UX
Questionnaires and results on device	***	**	Yes	Caters to requirement 2
Many question types	***	***	Yes	Relevant for standardized part
Record and play audio	***	**	Yes	Essential for qualitative interviews
Export results in CSV	***	*	Yes	Essential for this type of application
Import and export of files	**	**	Yes	Essential for this type of application
Question status	**	**	Yes	Considered useful for weakly structured interviews
Full Screen Mode	***	**	Yes	Essential for mixed method approaches
Location data	***	*	Yes	Caters to requirement 4
Multimedia in questions	*	***	No	No support from selected format queXML
Skip or hide question	**	***	No	No support from selected format queXML
Tags or bookmarks in audio	*	**	No	No direct support from used audio API
Question filters	*	*	No	Question status already visible with visual indication

6.3 Selected Features

Most of the features were already in some form part of the concept, and the user stories can be seen in appendix A, listed and marked if they were implemented. Together with the literature findings, the comparison of the applications, and by estimating complexities, features were selected for the prototype application. Another reason for this approach was the agile nature of development. After every iteration, the current implementation was tested and evaluated. This evaluation always had influence on the next steps and the course of the whole development effort. In table 6.1, all the features that were described in the previous section are together with their complexities, priorities, and if they found their way into the final prototype together with the reason. The priority for inclusion of a feature for the first prototype is not to be mistaken for the priority of such a feature in general in a survey application.

6.4 Implementation

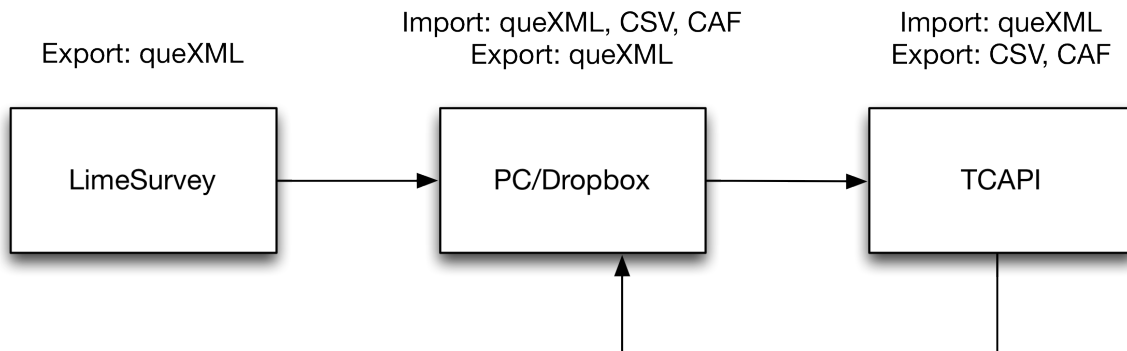
6.4.1 Mobile Platform

Two operating systems were considered for the creation of the prototype application. These two were Android from Google and iOS from Apple. Together they dominate most of the market of mobile devices as can be seen from the following passages. First, both systems are described shortly.

First released in 2007 for the first iPhone, the mobile OS iOS introduced many of the paradigms and interface elements that are widely adopted today in smartphones and tablets. With iOS, other innovations like multi-touch capable Liquid Crystal Display (LCD) displays and a centralized store for applications were introduced. Apple follows a vertical business model, which exclusively ties their software to the hardware that they produce. This favors the pace of application development as developers only have to consider so many devices and screen resolutions. The user base tends to update the operating system frequently, and the OS is currently in its seventh iteration. Native applications are written in Objective-C, an object-oriented superset of the popular C language, which is also the greatest entry barrier for developers as the conventions the language are unusual if developers come from other languages like Java and C++. However, this is mitigated through the excellent Integrated Development Environment (IDE) Xcode, also developed by Apple, which offers sophisticated tools for application development [*Apple Developer Program* 2013].

Android is a system that is based on Linux and applications are written in Java, which is also a very popular object-oriented language. The applications are compiled using the Android SDK. When running on a device, every application gets a unique Linux user ID from the operating system. With this approach, applications can be limited to access only resources and files that they need, which is called the *principle of least privilege*, therefore, creating a secure environment where applications cannot access critical system components [*Android Developer Homepage* 2013]. Each app is executed in a Virtual Machine (VM), and since version 2.2 Just In Time (JIT) compilation was introduced. With this approach the VM, called "Dalvik", increased execution time significantly [*Dalvik JIT* 2013]. Google follows a horizontal model with the Android OS, making it available to all interested hardware manufacturers. It is open source, and Google developed it with the Open Handset Alliance (OHA), which consists of over 80 technology companies [*Open Handset Alliance* 2013]. This is the greatest strength of the OS as it is free to anyone. This has made it the world's most popular mobile OS. On the other side, development for the platform becomes harder as a lot of devices, OS versions, and screen resolutions have to be considered when creating an application for Android. The development pace of the system is rapid, currently at version 4.4 "Jelly Bean", with the next version, 4.4 "KitKat" already announced.

The current situation of worldwide smartphone market share sees Android at 79% and iOS at 14.2% in the second quarter of 2013 [*Worldwide Smartphone Sales to End Users by Operating System in 2Q13* 2013]. Of course, this does not paint the whole picture for mobile devices market share as tablets are not included. However *Worldwide Devices Shipments by Operating System* [2013] published a report with worldwide PCs, tablets, and smartphones shipped, separated by OS. The prediction for the year 2013 sees Android leading with 36.9%, followed by Windows with 14.45%, iOS/Mac at 12.61% and RIM at 1%. The rest is others. In both categories it can be seen that Android is establishing itself as the leading platform, with at least

Figure 6.1: The ecosystem selected for the prototype.

twice as many devices shipped than the next competing OS, making it the system of choice if the target is to reach many devices. A similar trend can be seen when comparing only tablets. The market share of Android tablets rose from 38% in the second quarter of 2012 to 62.6% in the second quarter in 2013. During the same period iOS fell from 60.3% market share to 32.5% [IDC Worldwide Tablet Tracker, August 5, 2013 2013].

Nonetheless, during the survey application comparison the landscape of tablet optimized survey applications was perceived rather slim compared to iOS. Furthermore, because the focus of the TCAPI application is the evaluation of features which constitute a useful tool for interviews and surveys, iOS was chosen as a platform. The reasons for this were that there is only one resolution that the application needs to be optimized for and that the iOS System Development Kit (SDK) makes it convenient to create a tablet optimized interface. Finally, if a final version of the TCAPI application will get released, Android should strongly be considered as the main platform because of its ubiquity around the world and its rapid race to the top in both smartphone and tablet markets.

6.4.2 Ecosystem

As already mentioned in the sections above, queXML is a XML schema that defines questionnaires. LimeSurvey is a software tool for creating, analyzing, and exporting surveys. Although it would be greatly useful to create surveys directly on a mobile device, it was not deemed feasible for a prototype application. So the creation of questionnaires had to come from an external source, and LimeSurvey was chosen for this task. The ability to create complex questionnaires and export them in queXML, which is easy to understand and also open source, makes it a fitting solution for the problem at hand. The whole ecosystem can be seen in figure 6.1, where also the export features of the TCAPI prototype application can be seen. Questionnaires are created with LimeSurvey and exported in the queXML file schema. They are then imported in the application, either through a cable connection to a PC, or through a Dropbox download. Survey and interview results are exported in CSV files, and the audio recordings are exported in the .CAF format. Both can be exported through local storage and a tethered connection with a PC or a internet connection and Dropbox.

6.4.3 Database

While the questionnaire schema queXML is easy to understand and human readable, it is by no means simple. It contains numerous entities that describe various parts of a questionnaire, questions, and answers. To import a questionnaire into a device, the XML file had to be parsed. For this task, a Document Object Model (DOM) parser was chosen. It constructs the hierarchical structure of the XML and keeps it in memory. As already mentioned, queXML is very complex, making it not feasible to traverse the whole structure every time when a particular piece of information was needed. Hence, subparts of the UI do not need to know the whole structure. It can be easily imaged that a UI component for a question only needs to know about the question, and not the whole formation of XML nodes. So there was a mechanism needed to separate the whole system, making it more modular, which is most of the time considered a good practice in object-oriented design.

This is where CoreData comes into play, which is the API for object-oriented data storage in iOS. It creates an Object Relationship Model (ORM), and it is possible to have any technology beneath it to persist data. In this case, this was SQLite. By mapping the data structure to objects, only those that are needed are pulled out of the system which is more efficient than loading the whole structure every time. To do this, queries that are similar to those in other databases are used. By this approach, data is represented by objects together with all their relationships. So a question consists of an object with the question title as a property of it, in this example a string object.

The design of the database is a direct mapping of the queXML schema, making it very large with well over twenty entities. A subset of the entities can be seen in figure 6.2. This approach of course can be error prone, due to the sheer size of the DOM that is created from a questionnaire which needs to be mapped to the database with every containing data and relationships intact. So the approach to this task was to use TDD. Every node from the schema and its mapping to an object is covered by a unit test, making sure that all containing data and relationships are mapped correctly. Every node is tested separately, and at the end a large test is executed which calls everything recursively, effectively making this an executable proof that everything is imported correctly.

Furthermore, as questionnaire creation on the device is an important future feature, the foundation for it has been laid with this design approach. Not only can questionnaires be imported from queXML into the internal database, but export is also implemented on the database level. So a questionnaire from the database can be exported to a queXML file, which is also completely covered through unit tests.

6.4.4 User Interface

In this section, the final version of the TCAPAPI prototype created over the course of this thesis is shown. Different parts of the application are described, and in the section after it the implemented question types are listed together with the ones that LimeSurvey can export through queXML.

Home

When the application starts users are greeted with the home menu. The interface is kept simple, and all the imported questionnaires are listed as can be seen in figure 6.3. Questionnaires can

Figure 6.2: A part of the mapping of queXML to the CoreData database.

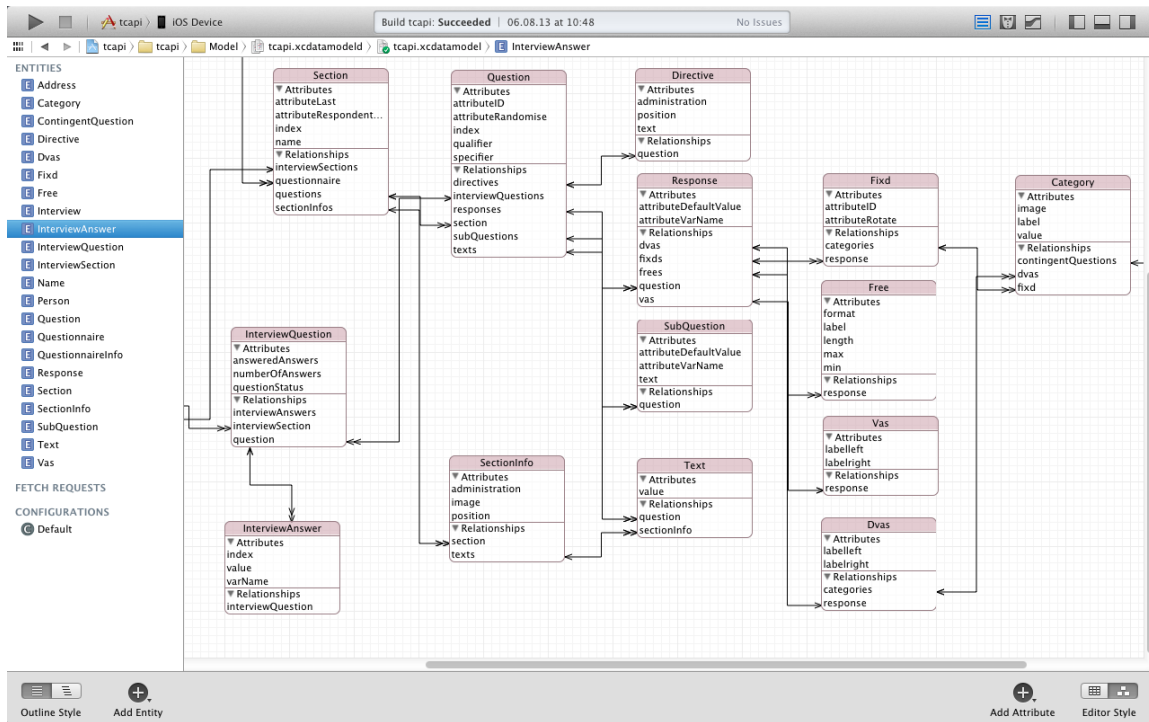


Figure 6.3: The home menu of the TCAPI prototype

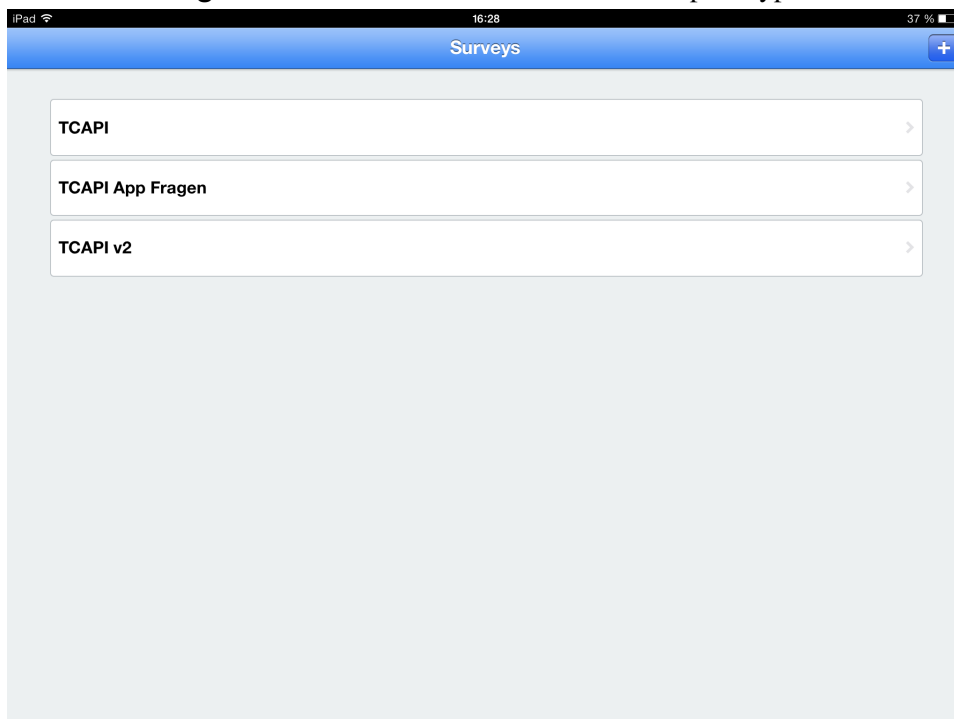
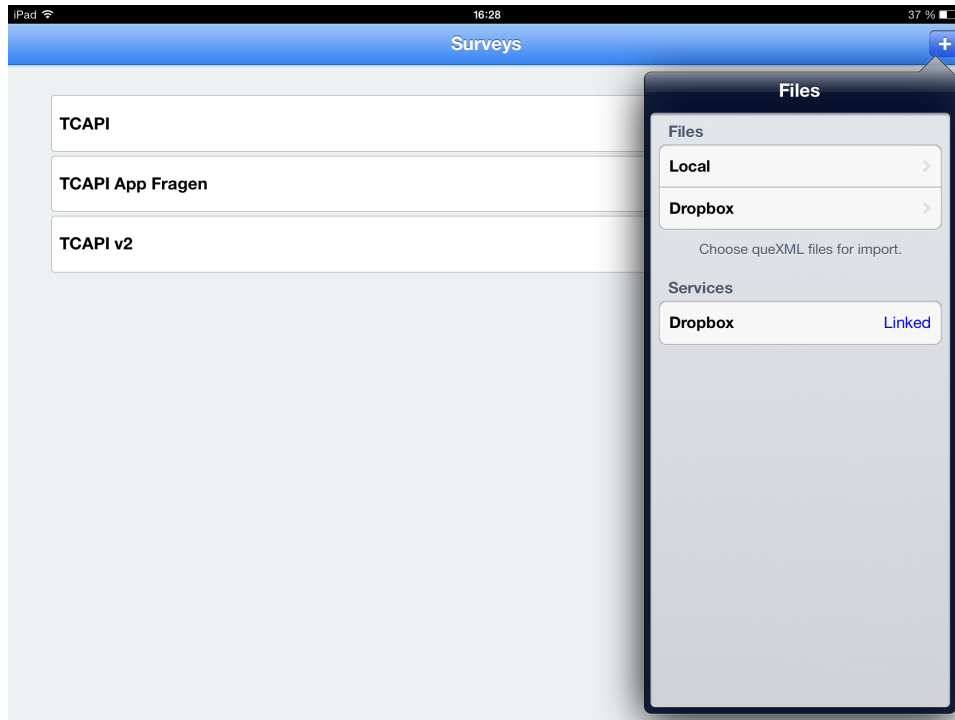


Figure 6.4: The menu where questionnaires can be imported.

be imported by pressing the "+" button in the upper right corner, which reveals a popover menu (Figure 6.4). Users can select either local files copied to the application by connecting it with the iTunes software, or by importing them through Dropbox [Dropbox 2013; Dropbox 2013]. To load files from Dropbox, a user has to link the application to the service by tapping it under the "Services" label. This opens a login window if no Dropbox application is installed, asking the user if TCAPi is allowed to access the service. However, if the application is installed on the device, no login is required, and the user is only shortly taken to the app to allow access. After this task, browsing through the online filesystem is possible.

For easier navigation, only folders and XML files are listed. When tapping a XML file, the questionnaire gets imported if its schema matches the queXML schema. Otherwise, a warning is displayed. No false data can be imported as every import is validated. By selecting a questionnaire from the list, the user gets to the next screen showing all the associated data.

Questionnaire

All previous surveys and interviews are listed in the questionnaire screen. They are separated in active interviews and saved interviews as can be seen in figure 6.5. Directly under the navigation bar are two selectable options. One being "Questionnaire", which shows the whole questionnaire without starting an interview, and the other being "New Interview", which, as the name suggests, creates a new active interview. In the top right is a "Share" button, from which the data export menu for this questionnaire and its associated data entries can be reached. Tapping any interview navigates to it and the current GPS location is persisted to the database.

Figure 6.5: Questionnaire menu with active and saved interviews.

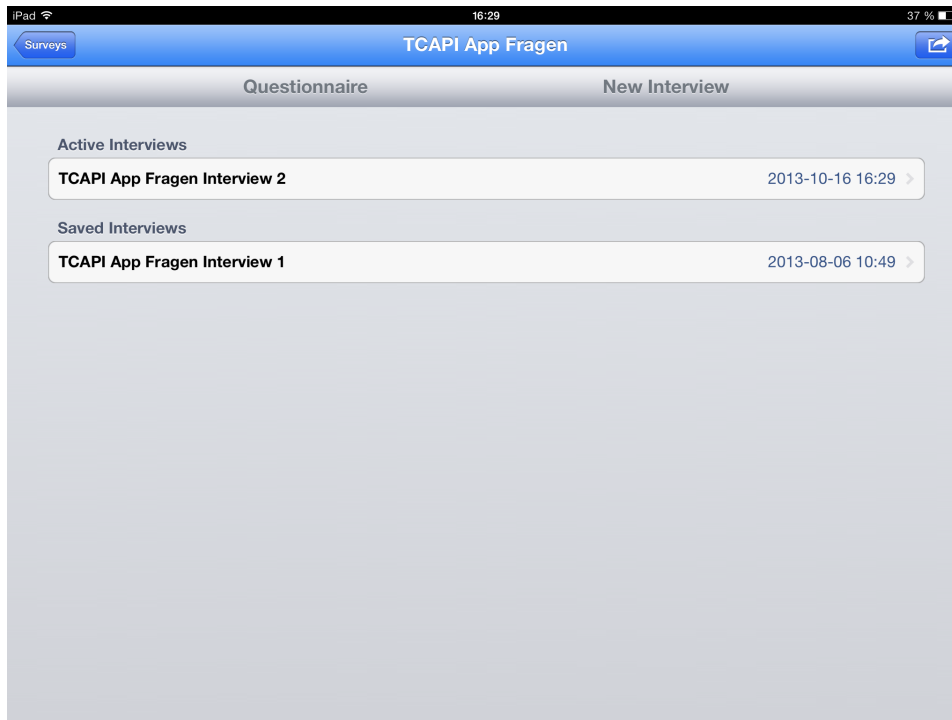
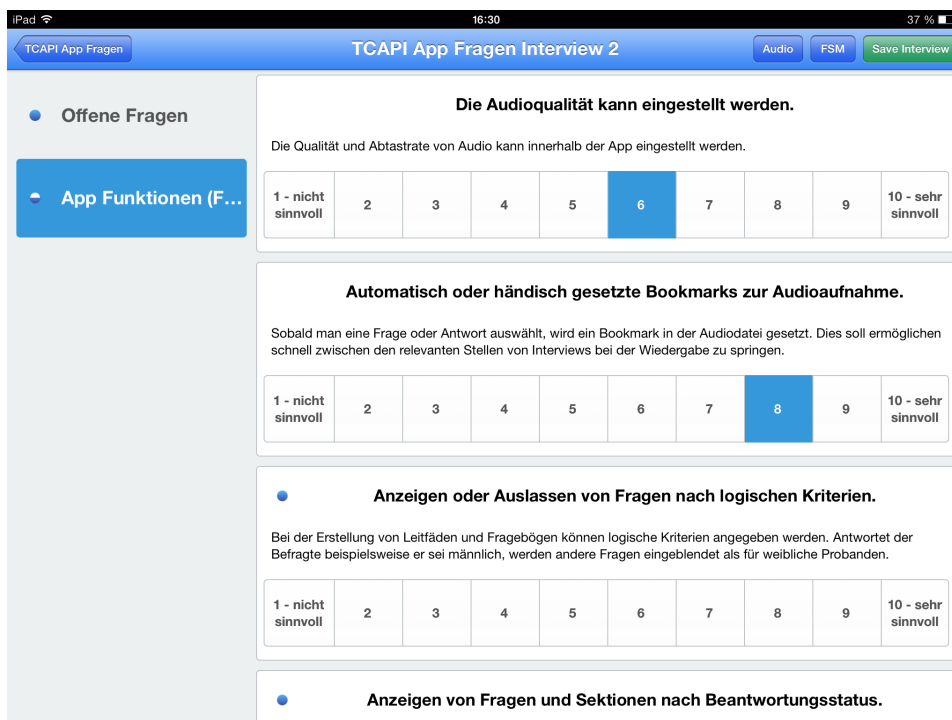


Figure 6.6: The interview screen with sections, questions, audio, FSM and question answer status indicators



Interview

The interview screen is busier than the previous ones, offering a large variety of options. Lets start with the questionnaire itself. Located at the left side is the list of sections. Selecting one of those items shows the accompanied questions, as can be seen in figure 6.6.

Questions consist of the question text, additional informal text, and UI elements for answering. In 6.6, there are only single choice selections. The selected item is clearly highlighted, and every answer can also be deselected, giving users full control over the answer status. This status is indicated by a blue circle for unanswered question, a half-filled circle for answered sections, and no circle if the question was answered, using the same metaphor that is used by email applications. This is not only implemented for single questions, but also for questions with multiple sub-questions and sections, giving the interviewer or self administered participant a quick glance over how much of the questionnaire was accomplished.

At the top right (navigation bar), there is a button to save and continue interviews. If this interview would be saved, changing the answers to questions would be disabled. This status is indicated by graying out the UI elements that do not offer any interaction anymore. This is also the mode in which already saved interviews or surveys are shown. With this approach, users get presented all the selected answers, but also all the other options, in contrast to other applications, which only show the selected data. Also, audio can be played for saved interviews, which brings us to the next feature on this screen. Furthermore, users can leave the application at any time and conveniently get back to it without losing data, as any change to any question is instantly persisted in the internal database. By supporting this, interviewers can switch to any application they want to and never lose their interview progress.

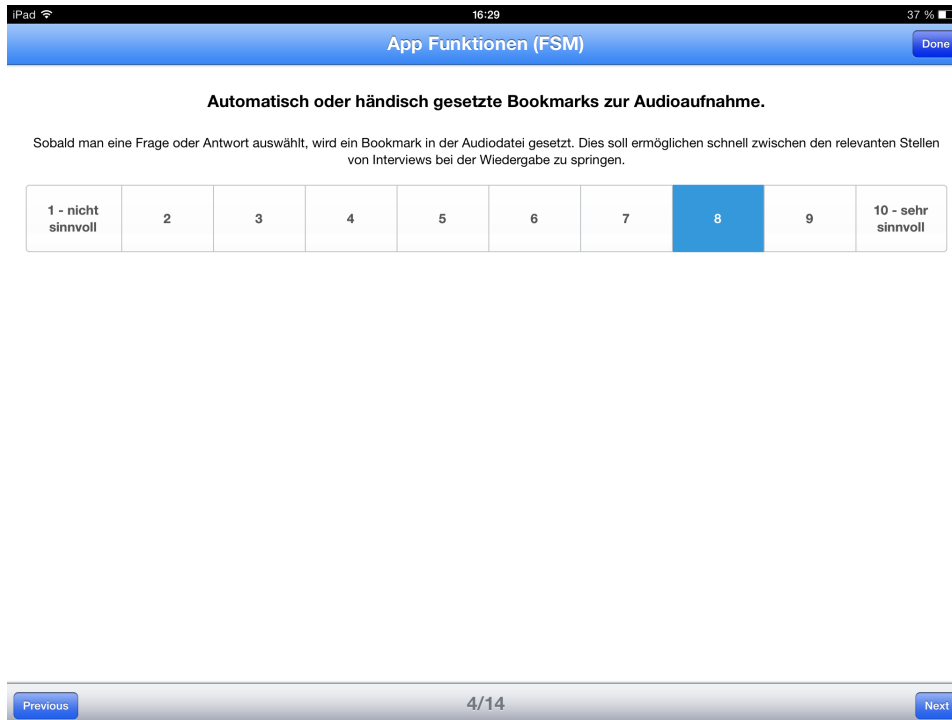
By tapping the "Audio" button on the top right, a popover is displayed. This popover offers two different kinds of controls. For an active interview, recording can be started and paused, and for a saved interview, the recorded audio can be played, if any is available. Like saving in the background, audio recording continues if the interviewer leaves the application. Even if the device is locked, the recording is not suspended. Audio is saved in the CAF file format. This is natively supported through hardware decoding and encoding, saving as much battery life as possible. Also, the standard profile that was chosen is configured for best clarity of voice recordings in conversations.

This leaves only one more button in this view, titled "FSM", which activates the Full Screen Mode.

Full Screen Mode

In the FSM, only one question at a time is displayed to the user. This mode is supposed to be used in situations where the interviewer hands over the tablet to the interviewee, where she/he can autonomously answer standardized questions. When pressing the "FSM" buttons, all the questions of a section are added, giving the interviewer the ability to separate interviews accordingly.

Consistent with the finding in the literature and the survey application comparison, navigation through questions is possible with "Next" and "Previous" buttons at the bottom. Also, the current question number with the number of all questions next to it is displayed as seen in figure 6.7. In addition to pressing the buttons for navigation, user can swipe left or right to change questions. If the navigation reaches an end in either direction (first and last question), the buttons get disabled and greyed out accordingly. Users can leave the FSM any time by pressing

Figure 6.7: The Full Screen Mode.

the "Done" button in the top right corner.

Other features like automatic question proceeding and having to provide an answer for proceeding have not been implemented for the prototype. Mainly because the interview situation is seen as a situation where both parties work together, reducing the need for additional control of the answer process.

Export

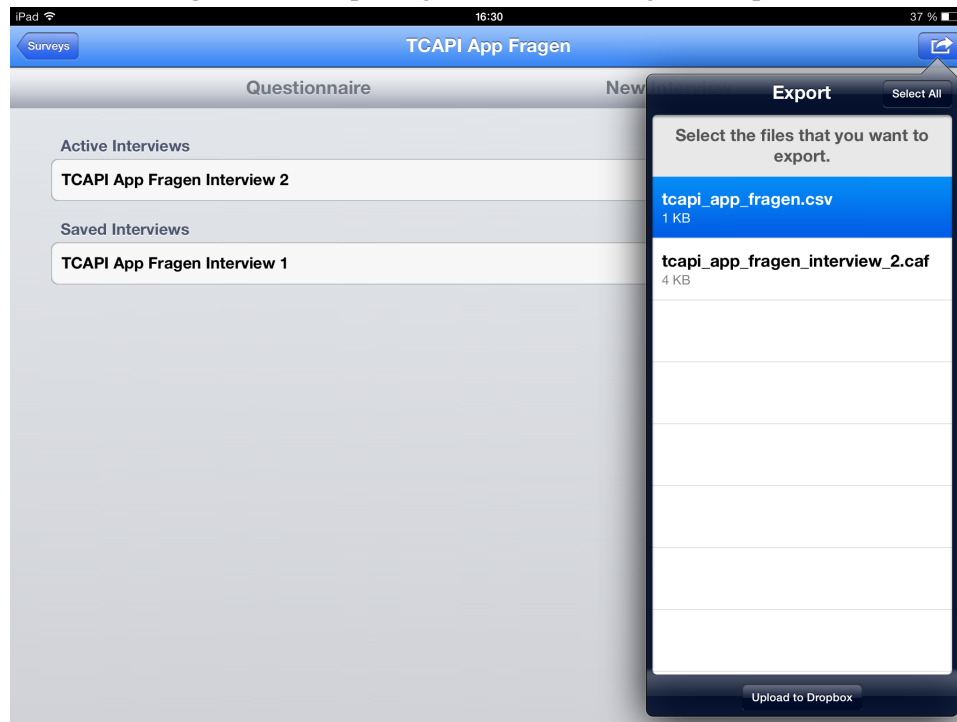
Users can share their results and audio through the questionnaire screen, like mentioned earlier. After pressing the share button, an overlay is displayed where files can be selected for export. This menu can be seen in figure 6.8. This functionality is for Dropbox export only. All the data is also saved through local storage, and can be exported using a tethered connection to a PC.

6.4.5 Question Types

Not all question types that LimeSurvey offers can be exported through queXML. In table 6.2, all the supported questions are listed and also which ones were implemented. The groups are array, mask, multiple selection, single selection and text questions.

6.5 Interviews - Field Study

Part of the evaluation of the TCAPI prototype are qualitative interviews about the practices that interviewers and researchers use during their interviews. Those are completed by Martin Griesbacher and Rafael Schögler at the Centre of Social Research at the Karl-Franzens-University

Figure 6.8: Exporting data and recordings to Dropbox.**Table 6.2:** Supported Questions

Question	Supported	Comments	User Interface
Array 5 point choice	Yes	-	Figure 6.9 a.)
Array 10 point choice	Yes	-	
Array Yes/Uncertain/No	Yes	-	
Array Increase/Same/Decrease	Yes	-	
Array Numbers	No	Placeholder shown	
Date	Yes	Date picker	Figure 6.9 b.)
Gender	Yes	-	
Numerical	Yes	Custom numerical pad	
Multiple Numerical	Yes	Custom numerical pad	
Yes/No	Yes	Not a drop down menu	
Multiple Options Checkbox	Yes	-	Figure 6.10 a.)
Dropdown	Yes	Not a drop down menu	Figure 6.10 b.)
5 Point Choice	Yes	-	
List Radio	Yes	-	
Multiple Short Text	Yes	-	Figure 6.10 c.)
Short Free Text	Yes	-	
Long Free Text	Yes	Same size as short text	
Huge Free Text	Yes	Same size as short text	

Figure 6.9: Implemented array and mask questions.

a.)

Array 5 point choice (A)

Row 1	1	2	3	4	5
Row 2	1	2	3	4	5

Array 10 point choice (B)

Row 1	1	2	3	4	5	6	7	8	9	10
Row 2	1	2	3	4	5	6	7	8	9	10

Array Yes/Uncertain/No

Row 1	Yes	Uncertain	No
Row 2	Yes	Uncertain	No

Array Increase/Same/Decrease

Row 1	Increase	Same	Decrease
Row 2	Increase	Same	Decrease

b.)

Date

Mittwoch, 09. Oktober 2013

7. August 2011
8. September 2012
9. Oktober 2013
10. November 2014
11. Dezember 2015

Multiple Numerical

Row 1	456
Row 2	853
Row 3	125

Gender

Female
Male

Numerical

Enter numeric value...

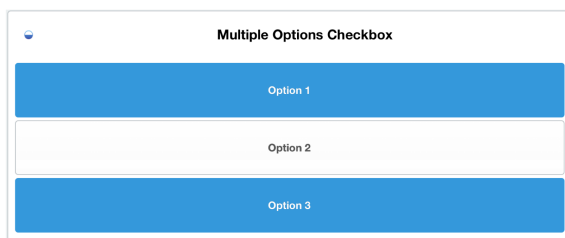
1	2	3
4	5	6
7	8	9
.	0	Delete

Yes/No

Yes
No

Figure 6.10: Implemented questions for single selection, multiple selection and text.

a.)



Multiple Options Checkbox

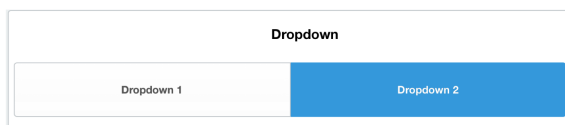
Option 1

Option 2

Option 3

This form displays three options. Option 1 and Option 3 are highlighted in blue, indicating they are selected. Option 2 is not highlighted.

b.)

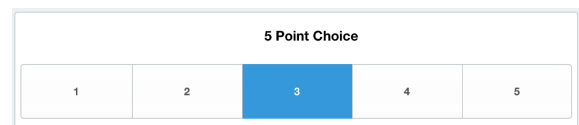


Dropdown

Dropdown 1

Dropdown 2

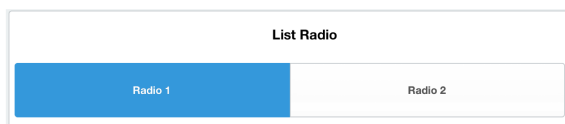
This form shows two dropdown options. Dropdown 2 is highlighted in blue, indicating it is selected.



5 Point Choice

1 2 3 4 5

This form shows five numbered options. Option 3 is highlighted in blue, indicating it is selected.



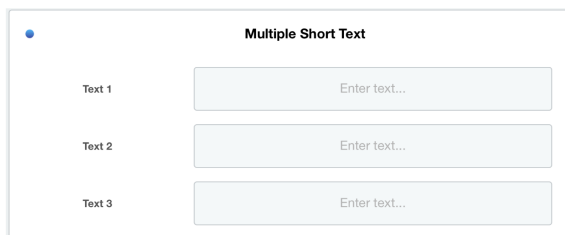
List Radio

Radio 1

Radio 2

This form shows two radio options. Radio 1 is highlighted in blue, indicating it is selected.

c.)



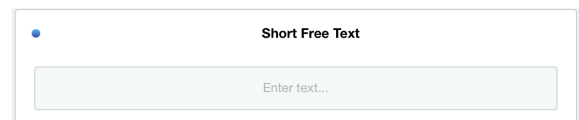
Multiple Short Text

Text 1 Enter text...

Text 2 Enter text...

Text 3 Enter text...

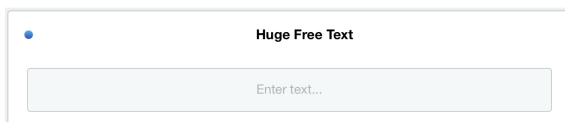
This form contains three short text input fields, each with a label (Text 1, Text 2, Text 3) and a placeholder "Enter text...".



Short Free Text

Enter text...

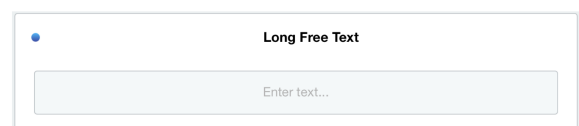
This form contains a single short free text input field with a placeholder "Enter text...".



Huge Free Text

Enter text...

This form contains a single huge free text input field with a placeholder "Enter text...".



Long Free Text

Enter text...

This form contains a single long free text input field with a placeholder "Enter text...".

of Graz, and they were kind enough to incorporate the standardized questions about the app features into their interviews and share the quantitative data that they have gathered. Questions were filled out autonomously by the participants using the FSM, and the TCAPI application was used for recording audio and also as a structured guideline for the interviews. It was divided into qualitative and quantitative parts, with the latter serving as an interview and feature evaluation.

The qualitative results are not part of the thesis, because the interview process it not completed yet. However, to get some insights into this part of the research, an expert interview with Rafael Schoegler was conducted.

6.5.1 Population

All together, nine people were interviewed, with 4 male participants and 5 female participants. Three participant answered that their main field of work is sociology. Other fields were social sciences, psychology, gender sociology, humane sciences, and social medicine.

6.5.2 Location

All interviews were performed at the same location, making this data point irrelevant when considering participants answers. Tying characteristics of their answers to the location was not possible. However, collecting location with the application was a feature was part of the standardized questions, and participants evaluated the usefulness of it.

6.5.3 Interview evaluation

Participants were asked about metadata as can be seen in table 6.3. The other questions regarded how participants conduct their interviews and what they thought about the use of the tablet computer during the interviews. The answers can be seen in table 6.4 and 6.5.

6.5.4 Feature Questions and Expert Interview

Part of the interview process was a quantitative questioning about current and possible future features of the prototype application. It was performed with the prototype application and the Full Screen Mode (FSM). Each following section will be dedicated to one such question and the resulting answers. In addition to the quantitative data, an expert interview with Rafael Schoegler was conducted, as he was one of the interviewers of the people involved in this research. Questions were open, and the standardized questions were used as a guideline. He always answered about the reactions of the participants while answering to those questions. Also, he gave his opinions on the data and the reactions of participants, offering insights into why this particular population answered in the way they did. The combination of this quantitative and explorative approach should help us to better understand the possible scenarios and adoption criteria for such a prototype among this population. For the expert interview, general rules of Mayer [2008] were followed.

The first question was about general criteria participants look for in mobile applications. For every following section, the question was about if the given function was considered mean-

Table 6.3: Question QI01 with subquestions

QI01: How useful would be the collection of following metadata in qualitative interviews?			
ID	Question	Likert Scale	Median
QI01a	location	very useful (1), somewhat useful, somewhat not useful, not useful (4)	2
QI01b	GPS-data	very useful (1), somewhat useful, somewhat not useful, not useful (4)	4
QI01c	date	very useful (1), somewhat useful, somewhat not useful, not useful (4)	1
QI01d	time of interview start	very useful (1), somewhat useful, somewhat not useful, not useful (4)	1
QI01e	time of interview end	very useful (1), somewhat useful, somewhat not useful, not useful (4)	1
QI01f	interview duration	very useful (1), somewhat useful, somewhat not useful, not useful (4)	1
QI01g	interviewee name	very useful (1), somewhat useful, somewhat not useful, not useful (4)	1
QI01h	contact information (telephone, email)	very useful (1), somewhat useful, somewhat not useful, not useful (4)	1

Table 6.4: Question QI02 with subquestions.

QI02: What of the following applies to your research: In my interviews...			
ID	Question	Likert Scale	Median
QI02a	I would need time to increase the number of interviews.	agree (1), somewhat agree, somewhat don't agree, don't agree (5)	2.5
QI02b	I would usually never ask standardized questions.	agree (1), somewhat agree, somewhat don't agree, don't agree (5)	3
QI02c	I have often forgotten to ask important questions.	agree (1), somewhat agree, somewhat don't agree, don't agree (5)	3
QI02d	I get confused with my interview guideline.	agree (1), somewhat agree, somewhat don't agree, don't agree (5)	4
QI02e	Sometimes my interview guideline is distracting me from listening.	agree (1), somewhat agree, somewhat don't agree, don't agree (5)	3
QI02a	Standardized questions disturb the willingness of the interviewee to talk.	agree (1), somewhat agree, somewhat don't agree, don't agree (5)	2
QI02f	Standardized questions would be interesting as a addition.	agree (1), somewhat agree, somewhat don't agree, don't agree (5)	2

Table 6.5: Qualitative questions without subquestions.

ID	Question	Likert Scale	Median
QI03	Thinking of your own interviews, how important is standardization or comparability?	important (1), somewhat important, somewhat not important, not important (4)	2
QI04	Tapping answers on the tablet screen felt more interesting than checking them on a paper questionnaire.	completely agree (1), somewhat agree, somewhat don't agree, don't agree (4)	3
QI05	Answering questions on the tablet computer was confusing.	completely agree (1), somewhat agree, somewhat don't agree, don't agree (4)	4
QI06	The tablet computer strongly disturbed the interview process.	completely agree (1), somewhat agree, somewhat don't agree, don't agree (4)	4
QI07	The tablet computer makes the interview more exiting.	completely agree (1), somewhat agree, somewhat don't agree, don't agree (4)	3
QI08	I thought that using the tablet computer during interviews was more of a toy than a serious research instrument.	completely agree (1), somewhat agree, somewhat don't agree, don't agree (4)	3
QI09	With the tablet computer to combine standardized and open questions in interviews.	completely agree (1), somewhat agree, somewhat don't agree, don't agree (4)	2
QI10	I don't think that tablet computer can be helpful for qualitative interviews.	completely agree (1), somewhat agree, somewhat don't agree, don't agree (4)	3

Table 6.6: Standardized questions about specific app features.

ID	Question	Mean	Median	Minimum	Maximum
QF02	Recording audio during interviews.	9.11	10	6	10
QF03	Setting the audio quality.	8.87	10	7	10
QF04	Automatic or manual setting of bookmarks for audio.	8.44	10	3	10
QF05	Show or skip questions with logic criteria.	8.11	10	5	10
QF06	Show questions and sections according to their answer status.	7.56	8	1	10
QF07	Handing over the tablet so that interviewees can answer standardized question.	7	7	3	10
QF08	Use of multimedia content (Audio/Video/Pictures) during an interview.	9.11	10	6	10
QF09	Automatic location tracking through GPS.	2	1	1	8
QF10	Export of standardized answers in a CSV data format.	9.44	10	6	10
QF11	Import and export of data through cloud services like Dropbox, Google Drive, or Box.	7	7	1	10
QF12	Creating interview guidelines or standardized questionnaires directly on the device.	8.11	8	5	10
QF13	Encryption of data for import and export.	8.63	9	6	10

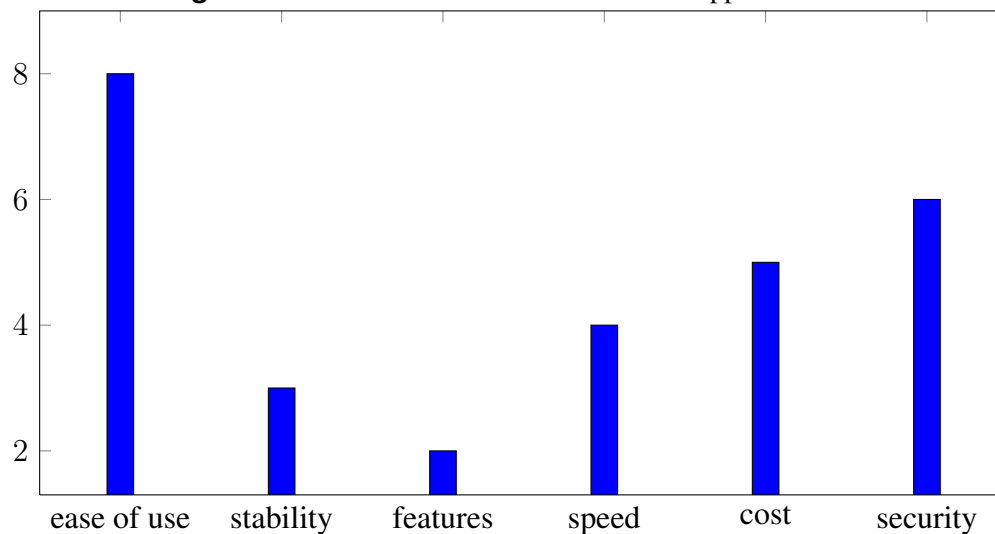
ingful. Participants could answer on a ten point scale, ranging from not useful to very useful. Table 6.6 shows all the results together.

QF01: What factors do you value the most in apps?

The possible answers were: ease of use, stability, features, speed, cost, and security. The most valued items to participants were ease of use and security as can be seen in figure 6.11. The term 'apps' required some explanation because the question was not clear to all participants. Almost all of the participants had experience with smartphones, but they had almost no experience with tablets. Nobody in the sample has bought apps for private use because, in their opinion, there are good enough free apps in almost every category. However, in regards of the TCAPi app, the expenses could be covered by universities and institutions for their employees. For students, an open access model was considered feasible.

QF02: Recording audio during interviews.

The mean value of the perceived usefulness of this feature is 9.11 with a median of 10. The minimum is 6 and the maximum 10. This was considered a very important feature of the TCAPi app as it is crucial to strongly qualitative interviews. Participants articulated a need for exceptionally good usability. They should be able to record, pause, interrupt and continue the recording process. The current time and status (Recording/ Not Recording) should be visible during all times of an interview. The audio should be playable on the computer for later transcription. Also, automatic transcription was mentioned as a desirable feature, because this task is very time-consuming. While there exists software for this very purpose, it is still very inaccurate.

Figure 6.11: Factors valued most in mobile applications.**QF03: Setting the audio quality.**

The mean value of this feature is 8.78 with a median of 10. The minimum is at 7 and the maximum at 10. This point was considered important but needed further explanation. As long as the speech is clearly understandable the quality of the audio recording was not considered to be important. However, by manipulating the quality of the recording, also the file size of the audio file changes, which can be a useful feature. One thing to consider is indicating graphically where the microphone is located on the device, for best results when orienting the tablet for recording. Also, the use of external, Bluetooth connected, microphones was considered as a possibly useful feature.

QF04: Automatic or manual setting of bookmarks for audio.

The mean for this feature is 8.44 with the median at 10 again. The minimum lies at 3 and the maximum at 10. The wording of this feature was not clear in the beginning. By manual bookmarks, it means small notes added to the audio recording through timestamps, and manually when a question or section is selected during the interview process. The later feature was seen as one with the greatest potential for innovation as it supports the transcription process by easily jumping through questions and section in the audio file. Only if the interviewer makes a large number of jumps between questions, and frequently goes back to previous sections, this feature could lose its usability. Nevertheless, this was positively received, but only if the timestamps are visible when playing the audio file on the computer and importing it in programs like MaxQDA [MaxQDA 2013].

QF05: Show or skip questions with logic criteria.

For the logic criteria, the usefulness was evaluated with a mean of 8.11 and the median at 10 again. The minimum was 5 and the maximum at 10. This was seen as useful, especially for strongly quantitative settings. Generally, this feature was seen as beneficial for long questionnaires and guidelines.

QF06: Show questions and sections according to their answer status.

The mean of this question lies at 7.56, making it the first feature which was perceived less useful in respect to the previously mentioned ones. The median is at 8, the minimum at 1 and the maximum at 10. With this feature, a kind of filter is incorporated to show the interviewer which questions are still open or which ones are answered. Also, showing all questions regardless of their status is possible. This was considered useful, under the condition that the interview guideline is somewhat longer. The standard procedure with paper based interviews is looking through all questions or notes at the end of an interview. Generally, participants saw the application as too much of an overhead if the interview guideline would be short. The use of this feature and the whole TCAPI app was more clear to them for longer interview situations and when teams have to work together.

QF07: Handing over the tablet so that interviewees can answer standardized question.

Again, a little worse with a mean of 7 and the median also at 7. The minimum was at 3 and the maximum at 10. This scenario is specifically considering the FSM (Full Screen Mode) of the application, which enables mixed mode methods. The population that was asked during the interviews did not find this feature overly useful. In interviews that they conduct, they tend to not mix the modes too much, especially in strongly qualitative settings. However, one scenario where this was considered useful is the gathering of sozio-demographical data like origin and family environment. During the expert interview with Rafael Schögler, he mentioned an idea from a conference he attended regarding this feature. It was proposed to use two tablets, essentially mirroring the questionnaire on both devices. In this scenario, both interviewer and interviewee can then look through questions together and answer them.

QF08: Use of multimedia content (Audio/Video/Pictures) during an interview.

Positively received with a mean of 9.11 and the median value at the maximum value 10. The minimum lies at 6 and the maximum at 10. The questioned sample tends to use only images or pictures during their interviews. Nonetheless, this is seen as a must for questionnaires on mobile devices as it has a tremendous potential for innovation if the screen size is large enough to support it.

QF09: Automatic location tracking through GPS.

The worst received feature with a mean of 2 and the median at the lowest possible value of 1. The minimum is at 1, and the maximum was 8. The interviewed population did not see large benefits with automatic GPS data collection in their fields of work. It was not obvious to them why they would need it and in which context they could use such a feature. For example, tracking interviewers as a means of control was not considered useful because the interviews would be of no use if the interviewers could not be trusted in the first place. However, several scenarios and contexts were mentioned where GPS might be useful. One such scenario are mobility surveys, where it is analyzed where and when people take which transportation. Also, location data could be used in city areas and shopping centers, where participants should describe what they see around them. In the current implementation, it is not possible to change the location or

deactivate it. The location is tracked at the start of an interview and never updated, making it less useful. These options were considered essential during the expert interview.

QF10: Export of standardized answers in a CSV data format.

Very good response with a mean of 9.44 and the median at 10. The minimum is 6 and the maximum 10. Most participants did not know the file format CSV, but understood the need for it immediately after explanation. This feature is absolutely essential for standardized questions.

QF11: Import and Export of data through cloud services like Dropbox, Google Drive, or Box.

This feature was somewhat lower scored with a mean of 7 and the median at 7. The minimum is 1 and the maximum is 10. Also, the reaction to this feature was strongly biased. There are strong reservations to using those services because of concerns regarding security and ethical considerations. Because of the sensible nature of research data, it is seen as a problem to save it in the cloud, essentially giving it in the hands of private corporations. Encryption was seen as a possible solution, and services like Wuala offer such possibilities. Generally, people are becoming more aware over security concerns regarding their data. For example, no one in the sample population uses Dropbox for sharing research data. Therefore, a strong need for tethered data transportation was expressed. When using cloud services, the data must be fully encrypted.

QF12: Creating interview guidelines or standardized questionnaires directly on the device.

This feature was rather positively perceived with a mean of 8.11 and the median at 8. The smallest value was 5, and the largest was 10. In the current implementation, questionnaires have to be created with LimeSurvey and imported. In general, participants noted that the more qualitative an interview is in its design the more of it should be possible to create on the device. Every functionality of the prototype application that is added from here on out should support this need. The focus of development should shift away from LimeSurvey as a tool for content creation, and go more to the device itself. This would be in the best interest of users as they do not have to have a LimeSurvey installation at home or at their institution. The more can be created on the device itself, the better.

QF13: Encryption of data for import and export.

The mean value lies at 8.63 and the median at 9. The minimum is 6 and maximum is 10. The answer of one participant was missing in the dataset. Like already mentioned, encryption is very important for the participants. This feature is seen as very important and should not be missing in a final version.

Random considerations about the prototype application.

During the qualitative interviews, information about more than just the aforementioned standardized questions and answers emerged. This was recorded during the expert interview with

Rafael Schögler and should also be incorporated into the decision about future version of the TCAPI application or similar ones.

The text size, although increased during the development process, was still considered too small by some participants. This could lead to problems for people with weak eyesight and should strongly be considered in further development efforts. For example, the text size could be set to be dynamic in height. The SDK of iOS 7 offers new text APIs that deal with exactly those cases. Moreover, other applications and platforms also offer dynamic sizing of text. Furthermore, the buttons for navigating through questions in the FSM (next and back) were perceived as too small. These targets should also be larger in future updates. These considerations can also be found in the literature and are of paramount importance for survey application design.

Another suggestion was the incorporation of more metadata for interviews and questionnaires and also making it editable. Furthermore, all files should be deletable. Not permitting this is especially problematic because of the size of the audio files, which can fill up the storage space of the mobile device very fast.

One suggestion was the incorporation of notes. All participants take notes in one form or another during their interviews. It was suggested to add a sidebar in the UI where notes can be written down. Another useful feature would be adding notes to particular questions. While considered useful in theory, the implementation of these features has to be very easy to use, reliable, and fast if it has to beat pen-and-paper notes. Generally, this applies to all features that require the attention of the interviewer. The less the interviewer has to concentrate on the device during interviews, the better. A high number of such features could be operated blindly by incorporating multi touch gestures. However, the problem with this approach is, that only advanced users know of them. For example, none of the participants used the swipe gestures incorporated in the FSM while answering the questions. The reason for this could also stem from the lack of familiarity with tablet computers in the interviewed population, and also that no explanation or visual hint was given to indicate this functionality.

Lastly, another suggested feature was drawing functionality within the application. At the beginning of interviews, especially with children, interviewers often give participants something to draw. Making such a functionality available in the application was considered desirable. Drawings could be saved digitally and exported, making this very useful for teams working together by having a complete digital data set with little effort. The same would be useful for hand written notes.

Chapter 7

TCAPI: Development Process and Implementation Details

“In the successful organization, no detail is too small to escape close attention.”

[Lou Holtz, American Football-Trainer]

A consistent development process and wisely chosen abstractions and frameworks are crucial to successful software development. Therefore, this chapter is dedicated to the details of the development process and of the prototype application which was build during the course of this work. It contains details about development, unit testing frameworks, SDKs, APIs, architectures, patterns and design decisions. This knowledge can be used by interested readers for planning and developing applications for interviews and questionnaires, but also in other fields. Furthermore, several frameworks were compared and alternatives listed.

7.1 Agile Development Process

7.1.1 Why Agile?

Software development has become increasingly complex over time. In the early days, when software was written in functional programming languages and applications were not overly complex, this was not as significant a problem as today. The problem with earlier approaches to development arose when software projects exploded in complexity and several people had to work on the same product. Functional programming would not cut it anymore, and only object-oriented languages could provide the scalability to react to these problems. With the latter, programs are not understood as a sequence of instructions on the macro level. They are modeled as objects interacting with each other by sending messages through well-defined public interfaces. Data that is only relevant for the internal workings of an object is kept private. Using this approach, software could be written as a composition of independent modules, which could be reused or replaced as needed.

However, not only the programming language is relevant for large scale projects, but also the development process. Usually, the waterfall model was used as depicted in figure 7.1 [Shore and Warden, 2009]. After collection requirements, the software is designed, then implemented

Figure 7.1: The development process according to the Waterfall model [*The Waterfall Model* 2013].

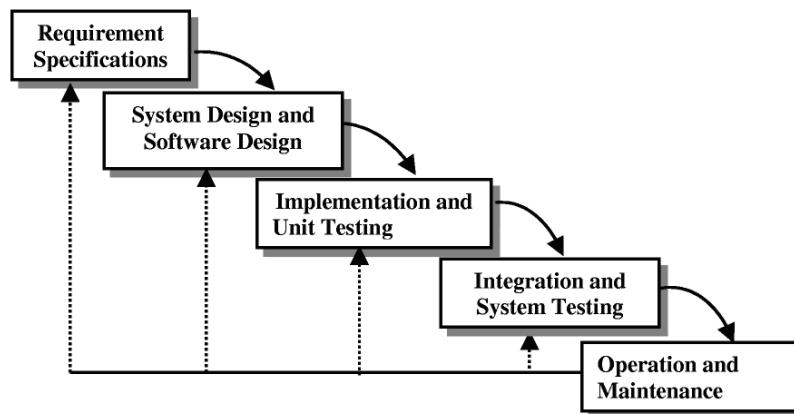


Fig. 1 Waterfall model

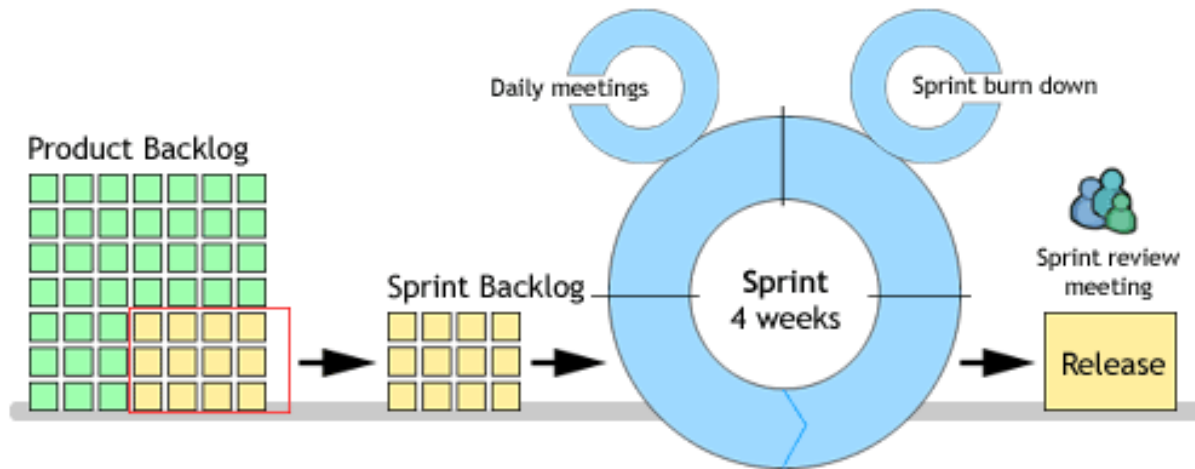
and finally verified and tested at the end. The process is not strictly iterative, and jumps to a previous step are possible. While this development process is very useful when it is exactly clear how the software should look like in the end, it becomes difficult to react to changes and new requirements. Also, due to the fact that testing is performed at the end of the cycle, software quality can suffer as this part is often neglected due to sharp deadlines. Typically, new features are implemented right until the end and testing is treated as an afterthought, resulting in unstable programs, and high costs if errors are detected late in the development cycle.

So how can agile methods address some of those problems? First of all, popular agile methods like Scrum and XP bet on TDD, where tests are written before the implementation. Also, a whole development cycle is drastically shortened by developing features in so called "sprints". A sprint usually lasts four to six weeks, depending on the needs of the projects. Projects are split into small chunks called stories, and only a predetermined number of stories are implemented per sprint [Shore and Warden, 2009]. Figure 7.2 shows the development process of the agile method Scrum. After each sprint, new stories, depending on their priority, are selected for the next sprint. These priorities can be altered throughout the development process, leaving it highly responsive to changes and new directions. All in all, by using agile methods, more stable software is produced through TDD, and software projects can respond to changing requirements and condition set by developers, managers or customers. The next sections will address more details about agile software development and its principles and inner workings.

7.1.2 Three Factors of Success

Still, agile is not a silver bullet to solve all problems in software development and at the same time increase productivity. For projects to be successful, [Shore and Warden, 2009] illustrate

Figure 7.2: The development process of Scrum [Agile Tools: Scrum 2013].



that organizational, personal, and technical success has to be met. Shore and Warden [2009] go on to describe how those goals can be met through incorporating agile principles in the development process:

Organizational Success One principle of agile methods is to include all stakeholders early in the development process. If, for example business people are involved early, the team can concentrate on features that deliver value to the organization, and they can work towards reducing the cost. This is done by continually improving and reducing complexities, both in code and process. Because of those aspects software becomes easier to maintain and extend.

Personal Success Personal success is achieved because all participating stakeholders can profit from agile methods. For example, developers gain more autonomy and also more control over deadlines and scheduling because they can estimate the complexities of individual tasks. This also gives team leaders and management more transparency, and their raised involvement in decision making is a large plus.

Technical Success Through principles like, TDD (Test Driven Development), coding standards, and Clean Code the quality, readability, and reusability of the produced code can be dramatically increased. The focus on tasks and finishing those before starting new ones greatly improves quality and reduces unexpected problems, which mostly lead to delays.

Agile methods like Extreme Programming (XP) are based on numerous concepts. One that was pervasive through the development process of the prototype was refactoring, or as Shore and Warden [2009] defines it:

"Refactoring is the process of changing the structure of code—rephrasing it—without changing its meaning or behavior. It's used to improve code quality, to fight off software's unavoidable entropy, and to ease adding new features."

This has tremendously helped at adding feature after feature to the product without scarifying code quality.

7.1.3 TCAPI: Agile Methods, Team, and Distribution

The above described methodologies were used as a guideline for the development process, and all involved parties interacted with each other on a regular basis. Beta versions of the TCAPI application were released in short cycles which lasted between two to four weeks. After every cycle, the current version was evaluated from all team members, and on this basis new priorities to the tasks at hand were assigned. With those, a new development cycle began. All in all, there were ten cycles and for the first two cycles issues and features were tracked through Jira, a software product for tracking issues and project management which also supports agile methods [Jira 2013]. With one sole developer and two people testing the beta versions, we learned fast that this approach while useful for larger or distributed teams, had too much overhead. After this realization, we switched to email to organize and evaluate the sprints. To distribute the application, TestFlight was used [TestFlight 2013]. With this service, other team members do not have to have Xcode to install the application as it can be distributed over the air.

Furthermore, to foster communication and make decisions about the UI more transparent, mockups were used. They are listed in appendix B.

7.2 Test Driven Development

When using TDD, software is written in a short cycle. When implementing a new feature test code is written first and executed so that it fails. After this, the corresponding production code is implemented. The minimal amount of code should be written that makes the test pass. Finally, when the test passes, this cycle is repeated. This cycle is how TDD is defined by Martin [2009, Page 77–79]. His exact definitions are the following three laws:

- Before writing production code, a failing unit test has to be written.
- Only so much code of the unit test should be written that it fails. Not compiling code also counts as a fail.
- The maximum amount of production code that should be written is the amount that the corresponding test passes.

By abiding to these simple rules, the cycle is very short. A test is written that fails, then a very small part of production code that just makes the test pass is written, and then everything is repeated.

It might take a programmer a while to get used to this kind of work-flow, but there are several benefits from this approach. When using it day after day, week after week, year after year, almost all of the *whole* codebase will be covered by tests. So when adding new functionality, all the tests can be executed and if they pass, almost certainly nothing was broken by the new code. This gives developers *confidence* and *certainty* in the changes they are making.

Another benefit is that the code is used as documentation. The test code shows how to use the system with all its function calls and object creations. This also means, that the test code is a *executable* specification of the whole system. Because if the test coverage is high, all the requirements should be verified through the tests.

When the three rules are followed, the production code is not even written in any form when the test is written. This means that the programmer has to think about the interface of the classes

and objects before they are written. But for them to work, they have to be testable, and therefore broken down into smaller units. By following the TDD approach, programmers are *forced* to think about decoupling their code, hence writing better software. [Martin, 2009, Pages 80–83]

7.3 Development Environment

Native applications for iOS are build with tools and an SDK which Apple provides to developers. Those are described in more detail in the following sections.

7.3.1 Distributing Applications

The OS iOS, formerly known as iPhoneOS, was introduced in 2007 with the release of the first iPhone. Shortly after the release followed the first SDK, with which developers could build applications and release them for the platform. Different from previous software distribution models, Apple introduced a centralized store where all applications are hosted. To release a software product in this store, developers have to abide by the rules that are established by Apple. These rules are called the App Review Guidelines, which have to be fulfilled to get an application accepted for distribution [App Review Guidelines 2013].

7.3.2 Xcode

Applications for iOS are, for the most part, developed using Apples IDE Xcode. It is a powerful tool with features for writing, compiling, and running code, version control, static code analysis, power and performance analysis tools, unit testing, and more. Also, it offers tools to build graphical user interfaces with a build in a drag-and-drop editor. The prototype was build with these tools.

7.3.3 Programming Language and SDK

The programming language used for iOS applications is Objective-C. It is, as the name suggests, an object-orient language build on top of the C language. It provides methodologies for higher abstractions with objects, classes, and methods, making it a modern language. Furthermore, it is a compiled language with manual memory management through reference counting. However, most parts of the memory management have been automated with the introduction of Automatic Reference Counting (ARC) with the introduction of iOS 5 in the fall of 2011. Releasing references is now performed by the compiler and only in edge cases does it have to be performed by the programmer. This makes writing applications for iOS both faster, less prone to errors through memory mismanagement, and execution is typically faster than on garbage collected languages like Java.

The iOS SDK offers various Application Programming Interfaces (APIs) for building applications. Be it the UI, APIs for persistence, security, audio, or networking. Of course, not everything is provided by the SDK. Third party developers also provide frameworks that can be incorporated into projects. Most of those are open source and can freely be used in commercial and non-commercial projects, when attributed accordingly. Some popular frameworks even replace the ones that are provided by Apple because they offer more features or simply are faster.

For example, the XML parser used in the TCAPI prototype was chosen because it offered a more convenient API and was also faster than the counterpart provided out of the box. A more detailed description is given in a later section.

Consequently, when choosing frameworks and API's, the one that best fits the needs of the product has to be chosen. The following sections are dedicated to the ones that were chosen for the prototype and why it makes sense to use them in the context of the TCAPI prototype application.

7.3.4 CocoaTouch: UIKit, Foundation, CoreData

Kochan [2011] defines a framework as:

”A framework is a collection of classes, methods, functions, and documentation logically grouped together to make developing programs easier.”

CocoaTouch is a collection of the most important frameworks of iOS. At least two of them, namely Foundation and UIKit, are used in every application written for iOS.

The Foundation framework provides objects for numbers, strings, and dates and also collection classes like arrays, dictionaries, and sets. Most of them have mutable and immutable versions. For example, once an NSArray with containing objects is instantiated, no more objects can be added or deleted. This is only possible with a NSMutableArray. It is almost identical to the Foundation framework provided on OSX, which is the operating system for Apple desktop computers and notebooks.

Another part of the Cocoa frameworks on the OSX side is Application Kit, which is a framework that provides user interface elements. Because of the different interaction paradigm of the desktop and mobile devices, this framework was not directly ported to iOS. The pendant to it is UIKit, which provides UI elements specially build for touch interfaces. Every class that has the prefix UI is part of this framework, and it is advisable to use those classes when building applications for iOS.

The most important ones are the UIViewController, which manages the lifecycle of a view that is displayed on the screen. Then there is UINavigationController, which is a controller containing controllers and manages the transition between them. There are a lot of other elements of UIKit that were used in building the TCAPI prototype, but describing every element is outside of the scope of this thesis. Instead, the two most relevant ones, UITableView and UICollectionView, are illustrated in the next sections.

Finally, there is CoreData, which is available on both OSX, as well as on iOS. It is a powerful framework for persisting data to the filesystem and accessing it through an object-oriented interface when needed. It shall be described in more detail in a later section. [Kochan, 2011]

7.4 Frameworks

With the previous section detailing the development environment and basic information about the SDK, this section will provide more detailed information about the frameworks that have been used, and how they work.

UITableView and UICollectionView

Nahavandipoor [2012, Pages 279–335] describes a `UITableView` as an interface element that is used whenever an application wants to display content in a list. The view is divided into sections, and those in turn are divided further into rows. For every row, a `UITableViewCell` object is returned. There are predefined cell objects, which can contain a title, subtitles, or an image. Furthermore, it is possible to create entirely custom cells by subclassing the cell class. The `UITableView` comes in two flavors, either normal or in a grouped style, where there is a greater separation between sections. Moreover, sections can have a header and a footer. While the `UITableView` is simple out-of-the-box, it is a very powerful UI element with extensive mechanism for extension and adaption.

So how does the mechanisms work with which the table is filled with data and events of it are reacted upon? Both are done through the use of protocols, the former with a *DataSource*, and the later with a *Delegate*. Both are objects that need to be set for the `UITableView`, either through code or with Interface Builder. The *Delegate* and *DataSource* both have to be classes that implement certain methods. Some of which are required, and others which are optional.

For the *DataSource*, three methods have to be implemented:

numberOfSectionsInTableView: Return how many sections there are in the `UITableView`. This method is optional, and the default value is one.

tableView:numberOfRowsInSection: Return how many rows are there in a section. The section number is a parameter of this method, beginning with the zero-index. This method is required.

tableView:cellForRowAtIndexPath: Return a cell for an index path, which consists of a row and a section. In this method, the cells must be instantiated and customized. This method is also mandatory to implement. However, one can imagine that the performance would suffer if for every row a new cell would have to be instantiated. This is especially true if there are hundreds or thousands of rows and cells. To counteract this problem, cells are reused when they go off the screen. That means that only the cells visible on screen are in memory.

By implementing those methods, the `UITableView` is provided with the data it displays. The *DataSource* also provides a means to edit operations like deletion and reordering. The *Delegate* methods on the other hand, provide a mechanism to respond to events like user touches, height changes of the cells, and information about headers and footers that the view needs to know. Some of the most common ones are [*iOS Developer Library* 2013]:

tableView:heightForRowAtIndexPath: Ask the delegate for the row height. This is called for every cell of the table view, so the calculation of cell heights should be performance optimized.

tableView:willSelectRowAtIndexPath: Respond to user touches. There is the method which begins with "will", called before the selection, and another with "did", called after selection. Both methods are also available for deselection.

tableView:viewForHeaderInSection: Ask the delegate for a header view. The same method is available for the footer. There are also methods for the titles and heights of header and footer views.

With iOS 6, UICollectionView was introduced. While it is similar to UITableView, also with sections and cells, it is not limited to displaying a list with horizontal scrolling. In fact, any custom layout can be realized. Grids, tiles, circular layouts, or pretty much anything the programmer can come up with is possible. For the TCAPI prototype, multiple collection views in table view cells were used. They were organized in a grid to display matrix questions with multiple radio or checkbox selections.

CoreData Framework

The CoreData framework provides an object-oriented interface for dealing with data persistence. It offers a higher abstraction for writing and reading data to files. This is the model part of the Model-View-Controller design pattern that is used extensively throughout the whole iOS system.

CoreData consists of several parts that work together. There is the NSManagedObjectContext, which defines relationships between NSManagedObject entities. Those can be compared to tables and rows that are used in databases like MySQL, where an entity is a table and attributes of the entity are table rows. CoreData offers an object-oriented interface, but underneath it are the files of the filesystem. How the data exactly is saved, should not be a concern of the programmer, as she or he is only working with the higher level APIs. It can be anything really, from files, to a SQLite database, or anything else. For example, for the unit tests the database was saved in memory instead of the filesystem. With this approach, testing all the separate units was fast. The bridge between physical files and the application is a class called NSPersistentStoreCoordinator. Another component is needed to add and retrieve data. With an NSManagedObjectContext entities can be created, loaded and deleted. Those changes are in memory and are only persisted if the "save:" method is called on the context.

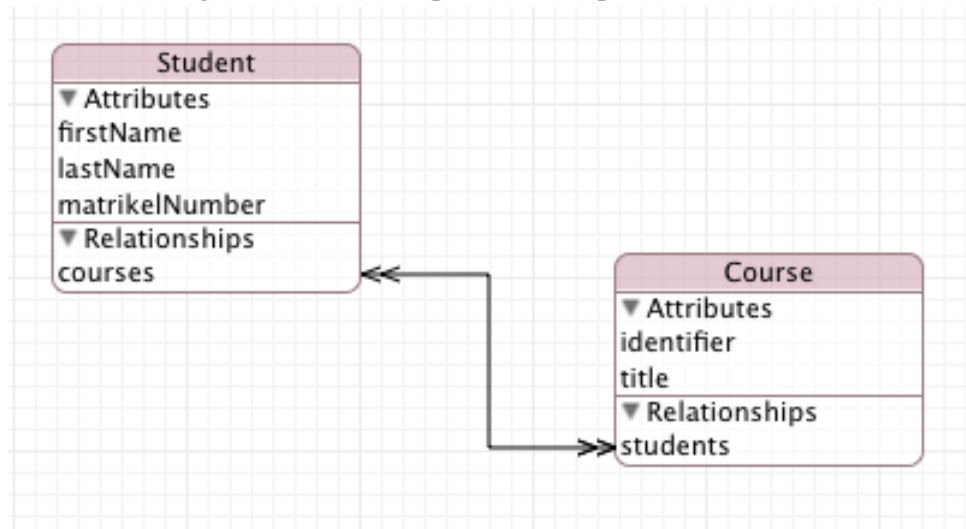
Like already mentioned before, a model describes NSManagedObject entities with attributes and their relationships to each other. Those relationships can be one-to-one, one-to-many, or many-to-many. For example, a student abstraction could be build, which has a many-to-many relationship with courses. Students visit many courses, and in every course there are, hopefully, many students. Attributes can have various types, like strings, numbers, and dates. In the above example, a student could have an unique number for identification called 'matrikelNumber', and strings for 'firstName' and 'lastName'. The relationship and attributes of this model can be seen in figure 7.3.

The code in listing 7.1 shows how to create and delete an entity. In this case, Student is a subclass of NSManagedObject. Such subclasses can be created automatically by Xcode from a model. After saving the context, the changes would get persisted to the filesystem.

```
1
2 Student *student = [NSEntityDescription
   insertNewObjectForEntityForName:@"Student"
   inManagedObjectContext:context];
3
4 [context deleteObject:student];
```

Listing 7.1: Creating and deleting a student within an context.

Fetching entities that are already stored in the database so that they can be displayed is done through a NSFetchRequest. The type of objects to fetch can be specified, together with

Figure 7.3: An example relationship between entities.

their ordering, and additional predicates which narrow the set of results. For example, in listing 7.2 all students who visit the course "Programming 101" are fetched and the result are sorted ascending by name. Since students can have multiple courses, 'ANY' is the predicate for courses with matching names. Finally "executeFetchRequest" is called on the context, and the results are returned as an NSArray.

```

1
2  NSFetchRequest *req = [NSFetchRequest fetchRequestWithEntityName:@"
   Student"];
3  req.sortDescriptors = @[ [NSSortDescriptor sortDescriptorWithKey:@"
   firstName" ascending:YES] ];
4  req.predicate = [NSPredicate predicateWithFormat:@"ANY courses.name
   == %@", @"Programming 101"];
5
6  NSArray *students = [context executeFetchRequest:req error:nil];
  
```

Listing 7.2: Fetching students.

When presenting data from a model to a user, things can get complex very fast. All data changes like insertion, deletion, and even updates have to be reflected in the UI. To cope with those, the class NSFetchedResultsController was introduced. Working in tandem with UITableView, it also has sections and rows. When hooked up to a UITableView the above mentioned changes are automatically reflected on the controller and consequently in the user interface. Also, cache can be managed more efficiently by only fetching a limited amount of objects in large datasets [Nahavandipour, 2012; *iOS Developer Library* 2013, Pages 663-693].

The TCAPI prototype makes extensive use of the CoreData framework with all the above described features. Questionnaires, sections, and questions are loaded and sorted with fetch requests. When an according predicate is set, different interviews for a questionnaire can be displayed. The UI is updated automatically if questions are answered. The whole data model is a mapping of the queXML standard, making the displaying, import, and access of different parts of the questionnaire fast, reliable, and simple.

CoreLocation Framework

The most prominent feature of mobile devices, as the name already suggests, is that they offer services to determine their location. Devices running iOS are not different, and the SDK provides the `CLLocationFramework` for this purpose. Even on devices which do not have GPS hardware build in, the location can be estimated pretty accurate if the device is connected to a wireless network [Nahavandipoor, 2012, Pages 434-437].

The TCAPI prototype records the location of the device every time a new interview is performed. An instance of `CLLocationManager` is created, and the current `ViewController` is set as delegate responding to the `CLLocationManagerDelegate`. The three most important methods are:

startUpdatingLocation This is a method of the `CLLocationManager`. It starts the tracking of location. If this is called the first time, the user is asked if he wants to permit the application to use location data.

locationManager:didUpdateToLocation:fromLocation Called when the location changes. The new location is given as a `CLLocation` object, and latitude and longitude can be accessed. The first time this is called in the TCAPI application, the coordinates for the current interview are saved, and location tracking is disabled again. This is a delegate method.

locationManager:didFailWithError: The location of the device could not be retrieved. React to this accordingly. This is also a delegate method.

AVFoundation Framework

The audio recording and playback functionality in the TCAPI app is implemented using the `AVFoundation` framework provided by Apple. It is highly configurable by setting the bit-rates, quality, number of audio channels, and different containers. Additionally, it also works in conjunction with other apps playing audio and in the background.

First, background audio is described. The SDK provides different categories to accommodate different scenarios where audio comes into play. The following categories can be selected as described in Table 7.1. The categories differ in several ways, namely, if the audio playback or recording can be silenced, other apps can play audio, and if playback, recording, both, or none is possible. For the purposes of the TCAPI application, the category `PlayAndRecord` was chosen as the app is able to continue recording, even if in the background or the screen of the device is locked. Also, for possible future version with media embedded, playing audio while recording is taking place is important. The category is activated by setting a key/value pair in the apps property list (PLIST) file [*Audio Session Programming Guide* 2013].

Regarding the audio format, CAF was chosen since it supports hardware encoding and decoding. This has several advantages. First, battery life is saved by using the hardware accelerated format and second, the CPU has to do less work. Software-encoded codecs would have to be encoded and decoded by the CPU, stealing resources that could be used for other tasks, like providing a fluent UI and user experience.

Since it is possible to record a high number of interviews, the amount of data that is saved on the device can be very large. To minimize this effect, a tradeoff between quality and size

Table 7.1: Audio Session Categories

Category	Silenced by Switch/Lock	Other App Audio	Playback	Recording
Ambient	Yes	Yes	Yes	No
SoloAmbient	Yes	No	Yes	No
Playback	Yes	No (Default)	Yes	No
Record	No	No	No	Yes
PlayAndRecord	No	No (Default)	Yes	Yes
AudioProcessing	-	No	No	No

has to be achieved. This can be done by reducing the channels, bitrates and quality of the audio stream.

Dropbox SDK

For the first couple of versions of the TCAPI prototype, PC's with the iTunes software were necessary to import questionnaires and export interview results. This was rather inconvenient and not what is expected from a truly mobile experience. Users want to have access to their files, wherever they are and not be tied to another machine. Therefore, a cloud service for file transfer had to be chosen. Dropbox is one of the most popular file hosting services with over 50 million users in 2012 according to Hunsinger and Corley [2012]. All people participating in the testing of the prototype had accounts and were familiar with the service.

Dropbox is a cloud service that makes sharing and synchronizing files and folders from and to numerous devices like smartphones, tablets, and personal computers convenient. Supported operating systems are Android, Blackberry, iOS, Windows Phone, Windows, OSX, popular Linux distributions, and others [Hunsinger and Corley, 2012]. While Dropbox is not the only cloud service of this kind (others include Google Drive, Microsoft SkyDrive, and Box), it offers a SDK (System Development Kit) that has a convenient API for iOS which can be used to asynchronously download and upload files using the delegation pattern. The SDK that was used was the Core API, and there is also the Datastore API and the Sync API. Those can be used for different kinds of applications or services and are described in the following way [Dropbox Developer 2013]:

Core API "The Core API provides a flexible way to read and write to Dropbox. It includes support for advanced functionality like search, revisions, and restoring files."

Sync API "Work with files on Dropbox through a familiar, file system-like interface. The Sync API takes care of syncing and notifying you of remote changes so your app can respond instantly. It also handles caching, network flakiness, and offline logic so you don't have to."

Datastore API "These days, your app needs to store and sync more than just files. With the Datastore API, structured data like contacts, to-do items, and game state can be synced effortlessly. Datastores work across platforms, offline, and even support automatic conflict resolution."

The Datastore API is clearly not suited for the need to download and upload files. While the Sync API is the easiest to use according to its description, it does not offer the flexibility needed in the prototype application. Therefore, the Core API was chosen.

Since users can upload multiple files with the TCAPI application and those can also be fairly large, a mechanism was needed to queue those files and upload them in the background, leaving the UI and the whole application responsive. Therefore, the whole functionality to upload files was encapsulated in a class with a protocol that is used to asynchronously inform an object that implements this protocol about the status of file uploads. The public interface and protocol definition can be seen in listing 7.3.

```

1
2 @interface EVODropboxUploadHelper : NSObject
3
4 - (void)uploadFile:(NSString*)filePath toDestinationPath:(NSString
   *)destinationPath;
5
6 - (void)startOrResumeUploads;
7 - (void)suspendUploads;
8 - (void)cancelAllUploads;
9
10 + (EVODropboxUploadHelper*)sharedInstance;
11
12 @end
13
14 @protocol EVODropboxUploadHelperDelegate <NSObject>
15
16 - (void)dropboxUploadHelper:(EVODropboxUploadHelper*)helper
   uploadProgress:(CGFloat)progress;
17
18 - (void)dropboxUploadHelper:(EVODropboxUploadHelper*)helper
   fileBeganUpload:(NSString*)filePath;
19 - (void)dropboxUploadHelper:(EVODropboxUploadHelper*)helper
   fileFinishedUpload:(NSString*)filePath;
20
21 @end

```

Listing 7.3: Dropbox Upload Class.

Files can be added by calling "uploadFile:toDestinationPath:" multiple times. When all files are ready, "startOrResumeUploads" needs to be called. All uploads can be suspended or cancelled altogether at any time. The protocol EVODropboxUploadHelperDelegate informs objects which implement it about the upload status. Also, files are not uploaded in whole, but in small chunks. This is necessary in mobile settings, where connections speed and signal quality are suspect to strong fluctuations. Also, possible, but not implemented in the current version, is reacting to connection issues and failed uploads. In this case, the file chunk that could not be uploaded will be uploaded again. This can be done either until it succeeds, or until a timeout is reached. In the latter case, the user has to be informed through an error message.

XML Parser

To import XML documents and traverse their structure, an XML parser was needed. There are two types of parsers, on the one side there are DOM parsers, which load the whole tree at once, and on the other side are Simple API for XML (SAX) parsers, which operate event based. The

Table 7.2: iOS XML Parser Performance

Name	Read	Write	Type	Download	Parsing	Total	API
libxml2 SAX	Yes	No	SAX	0.1223	0.0752	0.1438	C
NSXMLParser	Yes	No	SAX	0.0851	0.1500	0.2611	Obj-C
libxml2 DOM	Yes	Yes	DOM	0.0397	0.0951	0.1569	C
KissXMLParser	Yes	Yes	DOM	0.0444	0.1031	0.1722	Obj-C
GDataXMLParser	Yes	Yes	DOM	0.0431	0.1150	0.1833	Obj-C
TinyXMLParser	Yes	Yes	DOM	0.0503	0.2250	0.3014	C
TBXMLParser	Yes	No	DOM	0.0437	0.0540	0.1231	Obj-C
RaptureXMLParser	Yes	No	DOM	0.0437	0.0956	0.1636	Obj-C
TouchXMLParser	Yes	No	DOM	0.0462	0.1090	0.1787	Obj-C

latter is faster because it operates with events and call-backs and does not use large amounts of memory. However, with this serial fashion, the whole tree representation is available at once, which is arguably more convenient. So at the end, it is a tradeoff between speed and memory [Ray, 2001, Page 222]. As the whole structure of queXML needed to be mapped to a database, the tree representation was what was needed. Moreover, questionnaire documents tend to be short to medium sized, so the speed concerns for parsing were negligible.

Anyhow, the out the box offer from the iOS SDK was not suited for this because the NSXMLParser, which has an object-oriented API, is a SAX parser. The other option, lib2xml DOM, only has a C API. For this purpose, third party offerings were compared to each other, and a benchmark test comparing them was performed. For every parser, ten documents were parsed and the average speeds compared. This was done using a benchmark application on a third generation iPad, and the results can be seen in table 7.2 [*iOS XML Performance* 2013].

As read and write support was needed, and an object-oriented API was preferred, KissXML was chosen as the parser for the prototype application [*KissXML* 2013]. It is also relatively fast compared to its competitors and has almost double the performance than Apples offering. However, when speed is of stronger concern, a SAX parser should be considered for reading files, even though it is more complex to handle.

TPKeyboardAvoiding

When tapping a text field and entering text, the appearing keyboard often overlaps the content. Avoiding this in all instances of a complex interface is a rather tedious endeavor. This is where the third party extension TPKeyboardAvoiding comes into play, always guaranteeing that text input elements are visible when the keyboard is displayed [*TPKeyboardAvoiding* 2013].

MTStatusBarOverlay

Uploading and downloading files should always happen in the background and not block the Graphical User Interface (GUI) of an application. This becomes especially important with larger files, like audio recordings in the case of TCAPI. However, even if done in the background, it is still desirable to inform the user that the upload is happening and also show the progress of the operation. For this reason, MTStatusBarOverlay was incorporated. It shows status updates and progresses in the status bar of iOS devices, leaving the UI responsive and still providing progress information to the user [*MTStatusBarOverlay* 2013].

7.5 queXML

The open source project queXML is an XML schema which describes the structure of questionnaires. First lets start with describing the XML language and XML schemas. XML, short for EXtensible Markup Language, is a markup language that is used to transport and describe data. Therefore, it differs from the HTML markup language, which is used to display data. XML is self-descriptive and has no predefined tags, unlike HTML. That means that is is up to the user to define tags and the containing data. For example, listing 7.4 shows how a structure to describe a questionnaire, in this case a very simplified version of queXML, could look like. It describes a questionnaire with two sections, which have id's, and one question per section.

```

1 <questionnaire>
2   <title>Simple Questionnaire</title>
3   <section id="1">
4     <question>How old are you?</question>
5   </section>
6   <section id="2">
7     <question>What do you like about questionnaires?</question>
8   </section>
9 </questionnaire>

```

Listing 7.4: XML structure of a simple questionnaire.

The structure is a hierarchy of nodes with a tag. It contains other nodes and data like strings or numbers. The "questionnaire" tag is the root node, containing all the other ones. XML is a format which has to be well-structured, or else a XML parser would produce an error message [Vonhoegen, 2007]. Every node with an opening tag contained inside of the "<" and ">" bracket, also has to have a closing tag with a forward slash after the "<" symbol (e.g. "</tag>"). Like already mentioned, those tags are defined by the user in order to create custom data descriptions. However, not only the tags can be custom, also the whole hierarchical structure of an XML document can be defined. XML documents conforming to such a structure are then called valid document descriptions. To define this structure, either a Document Type Definition (DTD) or an XML schema has to be provided. XML files can be validated against those. Listing 7.5 shows a schema which defines that for every questionnaire there has to be exactly one title, at least one section, and for every section there has to be at least one question. Also, a section has to have an "id" attribute.

```

1 <xs:schema attributeFormDefault="unqualified" elementFormDefault="
2   qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
3   <xs:element name="questionnaire">
4     <xs:complexType>
5       <xs:sequence>
6         <xs:element name="title" minOccurs='1' maxOccurs
7           = '1' />
8         <xs:element name="section" minOccurs='1' maxOccurs
9           = 'unbounded'>
10        <xs:complexType>
11          <xs:sequence>
12            <xs:element name="question" minOccurs
13              = '1' maxOccurs='unbounded' />

```

```
10         </xs:sequence>
11         <xs:attribute name="id"/>
12     </xs:complexType>
13 </xs:element>
14 </xs:sequence>
15 </xs:complexType>
16 </xs:element>
17 </xs:schema>
```

Listing 7.5: A simple XML schema.

Chapter 8

Conclusion and Outlook

“It’s very easy to predict the future. People do it all the time. What you can’t do, is get it right.”

[Don Norman, The Front Desk, BBC Video, 1995.]

8.1 Conclusion

In chapter 4, mobile devices were seen in the context of social sciences. Their technologies, like touch screens, are widely researched and found to be feasible for surveys, and in many cases even better than their classic counterparts like pen-and-paper. For example, people found questionnaires on touch screens more secure, resulting in better quality answers for personal questions. Common between touch screen questionnaires was the UI, often showing one question at a time. Also, mixed method approaches were found in the research, as well as computer assisted interviews.

In chapter 5, applications for surveys, questionnaires, and interviews were compared to each other. The examined platforms were iOS and Android, and after an extensive search in their respective stores, eleven applications were selected for evaluation. After being evaluated, it was becoming clear that those applications are often linked to a web interface, acting as a client or companion application to a service. Functions like questionnaire creation and extensive reports are only available in the web interfaces, and not on the mobile device.

Also, similar to the research findings, applications that are used in strongly quantitative settings, have to offer an extensive library of question types and options for questionnaire design. Furthermore, no application was found that enables the combination of qualitative and quantitative methods. This leads to the conclusion that there is tremendous potential left in questionnaire and interview applications, especially when looking at content creation and mixing qualitative and quantitative methodologies.

The TCAPI project tries to live up to that potential with innovative approaches. With the project concept developed at the Karl-Franzens University of Graz and with Evolaris Next Level GmbH as the partner for software development, a prototype was developed. In chapter 6 the creation of this prototype was discussed. Together with the concept, findings from the research, and findings from the application comparison, requirements and features were derived. The latter were discussed in detail as well as evaluated, rated, and finally implemented. Those were

implemented and the final user interface presented. Due to the extensive need for hardware access, a native development approach was chosen.

Following this, the usage and potential of the prototype was evaluated through qualitative interviews that were conducted with the application, enabling the interviewer to use a mixed model approach by handing over the device for standardized questions. The most popular features amongst the participants were recording of audio, data export of results, and use of multimedia content like images, pictures, video, and audio. Also, creation of questionnaires on the device was seen as a desirable feature, and the largest potential for innovation was seen in automatically adding marks and notes to the audio recording.

Participants were strongly opposed to GPS collection in the background, and mentioned that they saw no situation in which they could use such a feature, not even for metadata collection. This is contrary to the findings in the literature and the application comparison, where location collection is universally used for many different scenarios. One possible explanation for this is, that this feature is not very useful in qualitative settings, but rather in quantitative ones. Furthermore, while participants found the connection to cloud services for data transfer useful, they articulated strong concerns regarding the security and privacy of such approaches. For example, none of them uses Dropbox and would also not consider using it for sensitive research data, and tethered connections were seen as a must.

More general questions regarding the application and its possible usefulness revealed positive results. The tablet computer was not found confusing or disturbing during the interview process. Participants could imagine it as a serious research instrument and thought that mixed model approaches could benefit from the use of tablet computers.

8.2 Outlook

The comparison of the applications in chapter 5 and the evaluation of the prototype in 6 revealed that there is a large potential for the tablet computer as a serious research tool for qualitative, quantitative, and mixed approaches.

However, several shortcomings of the TC-API prototype implementation were found. For example, the text size of questions and interview elements was found to be too small. While it was increased more than once during development, the text size was still perceived as too small. This becomes also more relevant when looking at the research, which reveals that especially elderly people have the need for larger fonts and legibility. Also, the dependence on other tools for questionnaire creation is another field that needs improving. The comparison also revealed a large number of features that are relevant for more complex questionnaire designs, like skips, jumps, randomization of answers, multimedia, and filters to only name a few. These need to be considered if the tool should offer stronger support to quantitative methodologies.

Finally, because of the need for features like audio recording, GPS capabilities, offline interviews and surveys, and extensive database support, building a native application was the right choice. However, if a wider reach of the application is the goal, other platforms have to be considered in the future.

Appendix A

TCAPI User Stories

This section is dedicated to the user stories of the TCAPI application. They are listed after a short description of the user roles. Comments are used to explain implementation details or why a story was changed.

AD Administrator. The person that created the survey or interview guideline.

INT Interviewer. The person performing the interview.

BEF Interviewee. The person that is interviewed.

Navigation		
Description	Comment	Implemented
As an INT i want to select between the options "Survey", "Creation", "Analysis", "Settings", "Info/About".	This was changed to a hierarchical model of navigation.	Modified
Survey		
Description	Comment	Implemented
As an INT I want to automatically see a list of recent questionnaires.	-	Yes
As an INT I want the list of questionnaires to be ordered chronologically.	Ordered alphabetically	Modified
As an INT I want to open a demo interview at the first start of the app.	Groundwork implemented. Can therefore be added easily.	No
As an INT I want to to continue previously started interviews (saving must be possible).	-	Yes
As an INT I want to have a "Share/Export" option.	-	Yes
As an INT I want to share data through tethered connections, email, or Dropbox.	Tethered and Dropbox	Yes
As an INT I want to delete interviews and questionnaires.	-	Yes

Interviewscreen		
Description	Comment	Implemented
As an INT I want to show and hide the list of questionnaire sections.	-	Yes
As an INT I want to start and stop the audio recording.	-	Yes
As an INT I want to add automatic marks to the audio when moving between questions.	-	No
As an INT I want to add manual marks to the audio.	-	No
As an INT I want the audio to be saved automatically so that it is not lost on interview abortion.	-	Yes
As an INT I want to have a "Sitemap" function with an overview over sections and questions.	-	Yes
As an INT I want the objects (questions) to be scrollable.	-	Yes
As an INT I want to scroll bar only to be visible while scrolling.	-	Yes
As an INT I want a automatic checkbox in the sections to indicate how many objects were answered.	-	Yes
As an INT I want the same automatic checkbox for questions.	-	Yes
As an INT I want the checkbox to be manually changeable (flagged, checked, unchecked).	-	No
As an INT I want a filter option for sections and questions with the options checked, flagged, unchecked, unchecked and flagged.	-	No
As an INT I want to have a status indicating how many questions have been answered.	Only in FSM.	No
As an INT I want to have automatically collect metadata (time, date).	-	Yes
As an INT I want to edit metadata.	-	No
As an INT I want to add interactive keyword to question objects which are color coded.	-	No
As an INT I want to add handwritten or typed notes.	Text question type can be created.	No
As an INT I want to switch from one question to another.	-	Yes

Full Screen Mode		
Description	Comment	Implemented
As an INT I want to enter the FSM with one klick of a button.	-	Yes
As an AD I want to edit wether upon answering another question is shown, a message is shown, or the screen is locked.	Screen locking not possible. Info question can be created.	Partly.
As an BEF I want to navigate through questions and see an info message at the end ("Hand over tablet to interviewer.").	Info message must be created by AD.	Yes
As an BEF I do not want to be thrown out of the FSM.	Not exactly specified what this contains.	Yes
As an INT I want to end the FSM even if not all questions were answered.	-	Yes
As an INT I want the audio recording to continue in the FSM.	-	Yes
As an INT/BEF I want to navigate through questions with arrows or swiping.	Next/Back buttons instead of arrows.	Yes

Creation		
Description	Comment	Implemented
As an AD I want to import questionnaires/sections/questions that were created on an PC/Mac.	-	Yes
As an AD I want the imported questionnaires/sections/questions to be sorted chronologically.	Questionnaires alphabetically. Others in order specified.	Yes
As an AD I want to set wether a question can be skipped in the FSM.	-	No

Analysis		
Description	Comment	Implemented
As an INT I want to see questionnaire chronologically ordered in analysis section.	Performed interviews are hierarchically under a questionnaire and not in separate analysis menu.	Modified
As an INT I want to see performed interviews with date, file size and name.	No file size as the data is separate in share menu.	Yes
As an INT I want to see metadata of interviews (time, location, contact information, etc.).	Only the title in app. Date and location in CSV export.	No
As an INT I want to edit the metadata.	-	No
As an INT I want to play the audio at the according mark when selecting a question object.	Audio for whole interview.	No
As an INT I want to export single/multiple/all interviews.	-	Yes

Settings		
Description	Comment	Implemented
As an AD I want to deactivate the sleep/energy saving function of the tablet.	Can be set in OS settings.	Yes
As an INT I want to activate Bluetooth microphones.	-	No
As an INT/AD I want to change the language (English/German).	-	No
As an INT/AD I want to see copyright information.	-	No
As an INT/AD I want to version information.	-	No
As an INT/AD I want to encrypt the audio files for sharing.	-	No

Appendix B

TCAPI Mockups

Mockups were created for easier communication during development. Figure B.1 and B.2 show the first set of mockups and figure B.3 the second set.

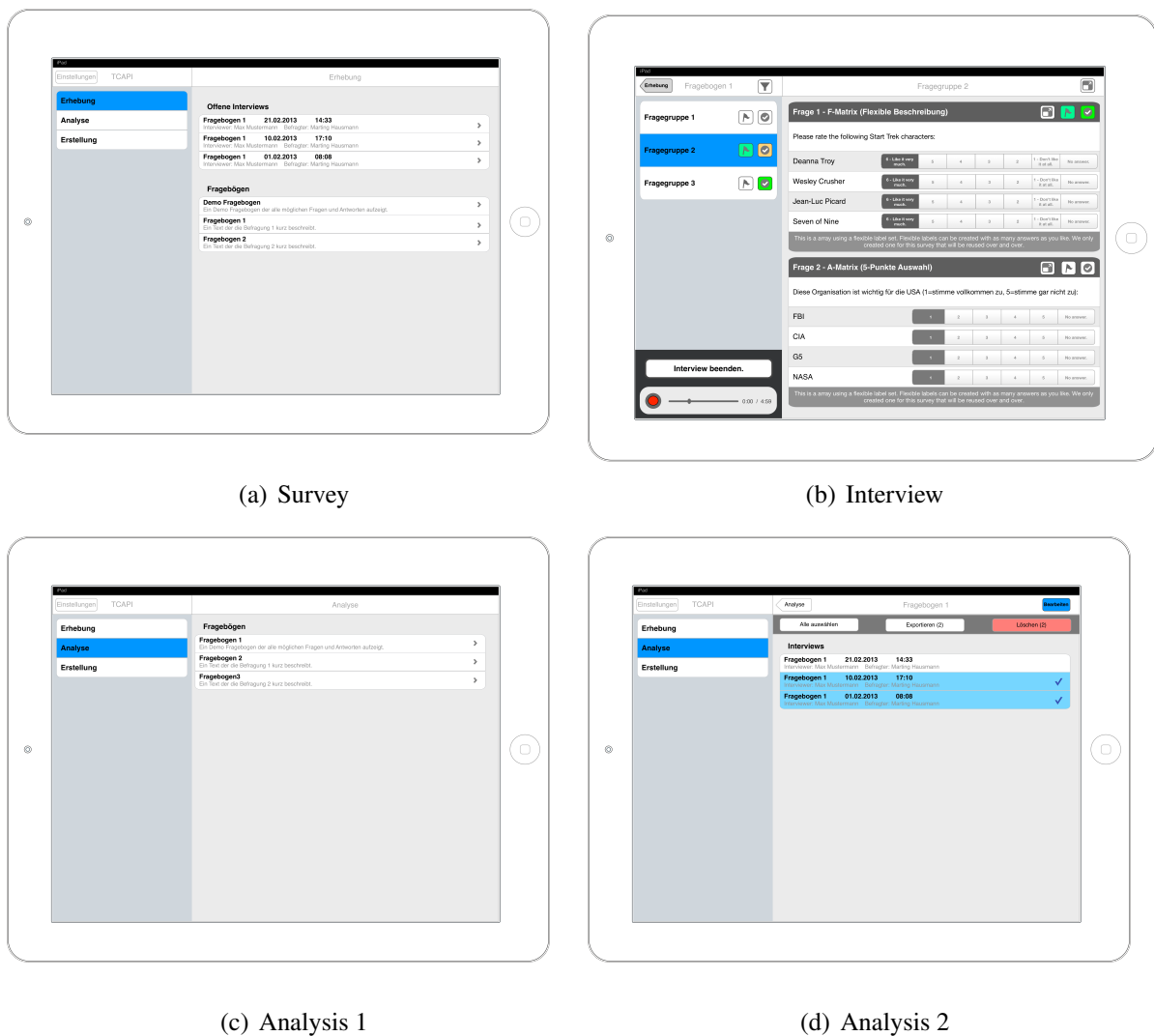
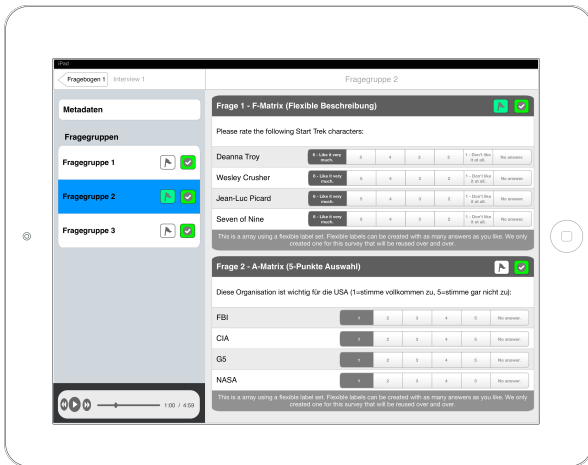
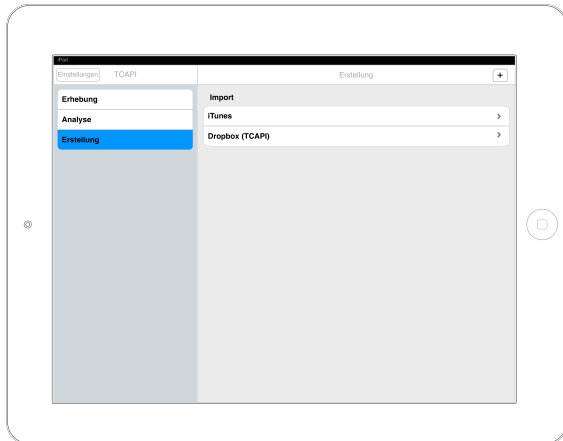


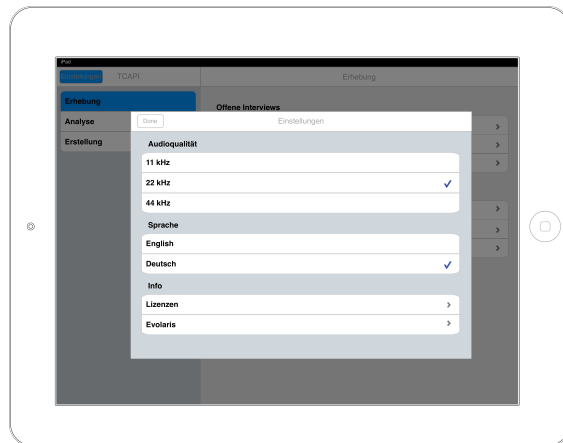
Figure B.1: First version mockups 1-4.



(a) Analysis 3

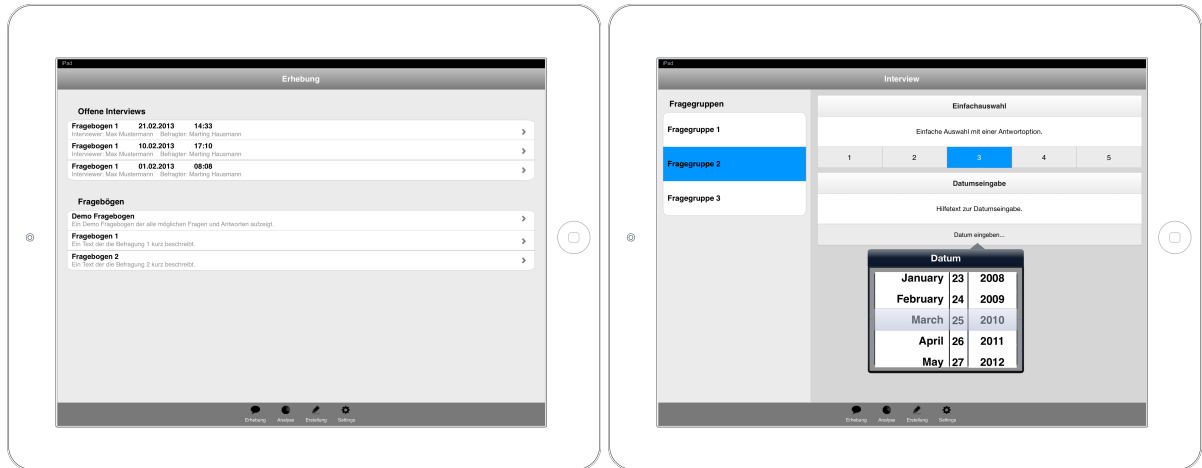


(b) Creation



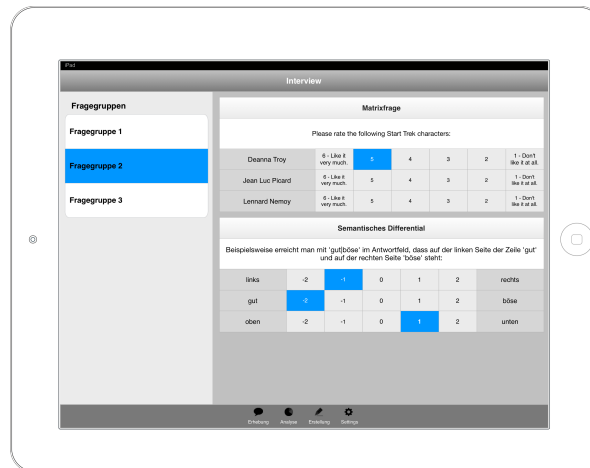
(c) Settings

Figure B.2: First version mockups 5-7.



(a) Survey

(b) Interview 1



(c) Interview 2

Figure B.3: Second version mockups.

List of Abbreviations

- ARC** Automatic Reference Counting
- API** Application Programming Interface
- CAPI** Computer Assisted Personal Interviewing
- CASI** Computer Assisted Self-administered Interviewing
- CATI** Computer Assisted Telephone Interviewing
- CSV** Comma Separated Values
- CPU** Central Processing Unit
- DOM** Document Object Model
- DTD** Document Type Definition
- FSM** Full Screen Mode
- GPS** Global Positioning System
- GUI** Graphical User Interface
- HIG** Human Interface Guidelines
- HRQOL** Health Related Quality of Life
- IDE** Integrated Development Environment
- JIT** Just In Time
- LCD** Liquid Crystal Display
- LBS** Location Based Services
- OHA** Open Handset Alliance
- ORM** Object Relationship Model
- OS** Operating System
- PAPI** Paper Assisted Personal Interviewing
- PC** Personal Computer

- PDA** Personal Digital Assistant
- SAX** Simple API for XML
- SDK** System Development Kit
- TCAPI** Tablet Computer Assisted Personal Interviewing
- TDD** Test Driven Development
- UI** User Interface
- UX** User EXperience
- VM** Virtual Machine
- WLAN** Wireless Local Area Network
- XML** EXtensible Markup Language
- XP** Extreme Programming

Bibliography

- Agile Tools: Scrum* [2013]. Oct. 12, 2013. <http://www.agile-tools.net/scrum.asp> (cited on page 65).
- Andrews, Keith [2012]. *Writing a Thesis: Guidelines for Writing a Master's Thesis in Computer Science*. Graz University of Technology, Austria. Oct. 22, 2012. <http://ftp.iicm.edu/pub/keith/thesis/> (cited on page xi).
- Android Design* [2013]. Mar. 28, 2013. <http://developer.android.com/design/index.html> (cited on page 37).
- Android Developer Homepage* [2013]. Mar. 28, 2013. <http://developer.android.com/> (cited on page 43).
- App Review Guidelines* [2013]. July 30, 2013. <https://developer.apple.com/appstore/guidelines.html> (cited on page 67).
- Apple Developer Program* [2013]. Sept. 22, 2013. <https://developer.apple.com> (cited on page 43).
- Armstrong, Just and Terry Overton [1977]. “Estimating Nonresponse Bias in Mail Surveys”. *Journal of Marketing Research* 14 (1977), pages 396–402 (cited on page 14).
- Atteslander, Peter [2010]. *Methoden der Empirischen Sozialforschung*. ESV Basics. Erich Schmidt Verlag, 2010. ISBN 3503126187 (cited on pages 9–11).
- Audio Session Programming Guide* [2013]. May 13, 2013. https://developer.apple.com/library/ios/documentation/Audio/Conceptual/AudioSessionProgrammingGuide/AudioSessionCategories/AudioSessionCategories.html#//apple_ref/doc/uid/TP40007875-CH4-SW1 (cited on page 72).
- Box* [2013]. Aug. 14, 2013. <https://www.box.com> (cited on page 24).
- Caviglia-Harris, Jill et al. [2012]. “Improving Household Surveys Through Computer-Assisted Data Collection Use of Touch-Screen Laptops in Challenging Environments”. *Field Methods* 24.1 (2012), pages 74–94 (cited on page 14).
- Computer and Internet Use in the United States* [2013]. Aug. 16, 2013. <http://www.census.gov/prod/2013pubs/p20-569.pdf> (cited on page 13).
- Cook, Colleen, Fred Heath, and Russel Thompson [2000]. “A Meta-Analysis of Response Rates in Web- or Internet-Based Surveys”. *Educational and Psychological Measurement* 60.6 (2000), pages 821–836 (cited on page 13).

- Dalvik JIT* [2013]. Oct. 10, 2013. <http://android-developers.blogspot.co.at/2010/05/dalvik-jit.html> (cited on page 43).
- DataField* [2013]. Sept. 17, 2013. <https://www.data-field.com> (cited on page 23).
- Dawson, Alexander et al. [2012]. *Essentials of Mobile Design*. 1st Edition. Smashing Media GmbH, 2012. ISBN 9783943075427 (cited on page 4).
- De Bruijne Arnaud, Marika and Wijnant [2013]. “Comparing Survey Results Obtained via Mobile Devices and Computers: An Experiment With a Mobile Web Survey on a Heterogeneous Group of Mobile Devices Versus a Computer-Assisted Web Survey”. *Social Science Computer Review* (2013) (cited on page 18).
- DeBree, R et al. [2008]. “Touch Screen Computer-Assisted Health-Related Quality of Life and Distress Data Collection in Head and Neck Cancer Patients”. *Clinical Otolaryngology* 33.2 (2008), pages 138–142 (cited on pages 14, 15).
- Denny, Simon J et al. [2008]. “Hand-Held Internet Tablets for School-Based Data Collection”. *BMC Research Notes* 1.1 (2008), page 52 (cited on page 16).
- Dillman, Don A. et al. [2009]. “Response Rate and Measurement Differences in Mixed-mode Surveys Using Mail, Telephone, Interactive Voice Response (IVR) and the Internet”. *Social Science Research* 38.1 (2009), pages 1–18. ISSN 0049-089X. doi:10.1016/j.ssresearch.2008.03.007 (cited on page 13).
- Dropbox* [2013]. Sept. 22, 2013. <http://www.dropbox.com> (cited on page 47).
- Dropbox* [2013]. Sept. 22, 2013. <http://www.apple.com/at/itunes/download/> (cited on page 47).
- Dropbox Developer* [2013]. Sept. 19, 2013. <https://www.dropbox.com/developers> (cited on page 73).
- Dupont, Alexandra et al. [2009]. “Use of Tablet Personal Computers for Sensitive Patient-Reported Information”. *J Support Oncol* 7.3 (2009), pages 91–97 (cited on pages 16, 17).
- Dutzler, Roland, Martin Ebner, and Robert Brandner [2013]. “Indoor Navigation by WLAN Location Fingerprinting-Reducing Training-Efforts with Interpolated Radio Map”. In: *UBI-COMM 2013, The Seventh International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*. 2013, pages 1–6 (cited on page 4).
- Ebner, Martin, Christian Stickel, and Josef Kolbitsch [2010]. “iPhone/iPad Human Interface Design”. In: *HCI in Work and Learning, Life and Leisure*. Springer, 2010, pages 489–492 (cited on page 4).
- Fann, Jesse R et al. [2009]. “Depression Screening Using the Patient Health Questionnaire-9 Administered on a Touch Screen Computer”. *Psycho-Oncology* 18.1 (2009), pages 14–22 (cited on pages 14, 15).
- Findlater, Leah, Jacob O Wobbrock, and Daniel Wigdor [2011]. “Typing on Flat Glass: Examining Ten-Finger Expert Typing Patterns on Touch Surfaces”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. 2011, pages 2453–2462 (cited on page 17).

- Flick, U, E von Kardorff, and I Steinke. *Qualitative Forschung - Ein Handbuch*. 7th Edition. Hamburg, Germany: Rowohlt Taschenbuch Verlag. ISBN 3499556289 (cited on page 10).
- FluidSurveys* [2013]. Sept. 17, 2013. <http://fluidsurveys.com> (cited on page 28).
- Fuchs, Marek, Mick Couper, and Hansen SE [2000]. “Technology Effects: Interview Duration in CAPI and Paper and Pencil Surveys”. *Developments in Survey Methodology*. Ljubljana, Slovenia (2000), pages 149–166 (cited on page 15).
- Human Interface Guidelines* [2013]. Mar. 28, 2013. <https://developer.apple.com/library/ios/documentation/userexperience/conceptual/mobilehig/> (cited on pages 4, 37).
- Hunsinger, Scott D and Ken J Corley [2012]. “An Examination of the Factors Influencing Student Usage of Dropbox, a File Hosting Service”. In: *Proceedings of the Conference on Information Systems Applied Research ISSN*. Volume 2167. 2012, page 1508 (cited on pages 39, 73).
- IDC Worldwide Tablet Tracker, August 5, 2013* [2013]. Oct. 13, 2013. <http://www.gartner.com/newsroom/id/2573415> (cited on page 44).
- iOS Developer Library* [2013]. Aug. 19, 2013. <http://developer.apple.com/library/ios/navigation/> (cited on pages 69, 71).
- iOS XML Performance* [2013]. Feb. 3, 2013. <https://github.com/jansanz/iOSXMLPerformance> (cited on page 75).
- iSurvey* [2013]. Sept. 17, 2013. <https://www.isurveysoft.com> (cited on page 26).
- Jira* [2013]. Jan. 3, 2013. <https://www.atlassian.com/de/software/jira> (cited on page 66).
- Kelle, Udo [2008]. *Die Integration qualitativer und quantitativer Methoden in der empirischen Sozialforschung: Theoretische Grundlagen und methodologische Konzepte*. Springer DE, 2008 (cited on page 10).
- KissXML* [2013]. Feb. 3, 2013. <https://github.com/robbiehanson/KissXML> (cited on page 75).
- Klemmerer, Tobias [2011]. “Umfragesysteme auf Basis mobiler Endgeraete - Untersuchung der Mehrwerte gegeneuber papierbasierten und Online Frageboegen auf stationaeren Gerueten”. Master’s thesis. Goethe Universitaet, Frankfurt am Main, Germany, 2011. <http://www.webzwerk.net/tobiaskemmerer/DiplomarbeitMobileUmfragesystemeXXL.pdf> (cited on pages 17, 18).
- Kochan, Stephen [2011]. *Programming in Objective-C*. 4th Edition. Addison Wesley, 2011. ISBN 0321811905 (cited on page 68).
- Krug, Steve [2006]. *Don’t Make Me Think! a common sense approach to web usability*. 2nd Edition. New Riders, 2006. ISBN 9780321344758 (cited on pages 36, 37).
- Larsson, Bodil Wilde [2006]. “Touch-Screen Versus Paper-And-Pen Questionnaires: Effects on Patients’ Evaluations of Quality of Care”. *International Journal of Health Care Quality Assurance* 19.4 (2006), pages 328–338 (cited on pages 14, 16).

- Leeuw, Edith D De [2005]. “To Mix or Not to Mix Data Collection Modes in Surveys”. *Journal of Official Statistics* 21.5 (2005), pages 233–255 (cited on page 11).
- LimeSurvey: Open Source Survey Creation and Distribution* [2013]. Apr. 3, 2013. <https://www.limesurvey.org> (cited on page 38).
- Loop Survey Maker* [2013]. Sept. 17, 2013. <http://www.loopsurvey.com> (cited on page 21).
- Manfreda, Katja Lozar et al. [2008]. “Web Surveys Versus Other Survey Modes: a Meta-Analysis Comparing Response Rates”. *Journal of the Market Research Society* 50.1 (2008), page 79 (cited on page 13).
- Martin, Robert C [2009]. *Clean Code: A Handbook of Agile Software Craftsmanship*. 1st Edition. Prentice Hall, 2009. ISBN 9780132350884 (cited on pages 66, 67).
- MaxQDA* [2013]. July 7, 2013. <http://www.maxqda.de> (cited on page 58).
- Mayer, Horst O [2008]. *Interview und schriftliche Befragung: Entwicklung, Durchführung und Auswertung*. 4th Edition. Munich, Germany: Oldenbourg Verlag, 2008. ISBN 3486586696 (cited on page 54).
- mQuest Surveys* [2013]. Sept. 17, 2013. <http://www.mquest.eu/survey-app/overview/> (cited on page 20).
- MTStatusBarOverlay* [2013]. Apr. 13, 2013. <https://github.com/myell10w/MTStatusBarOverlay> (cited on page 75).
- Nahavandipoor, Vandad [2012]. *iOS 6 Programming Cookbook*. 1st Edition. O’Reilly Media, Inc., 2012. ISBN 1449342752 (cited on pages 69, 71, 72).
- Nguyen, Bobby T and Barbara S Chaparro [2012]. “Apple iPad Usage Trends by Students and Non-Students”. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Volume 56. 1. SAGE Publications. 2012, pages 1511–1515 (cited on page 17).
- Open Handset Alliance* [2013]. Oct. 10, 2013. <http://www.openhandsetalliance.com> (cited on page 43).
- Opinionmeter Touchfield* [2013]. Sept. 17, 2013. <http://opinionmeter.com> (cited on page 26).
- Qualtrics Surveys* [2013]. Sept. 17, 2013. <http://qualtrics.com> (cited on page 31).
- queXML: Open Source XML Questionnaire Schema* [2013]. Apr. 3, 2013. <https://www.limesurvey.org> (cited on page 38).
- Ray, Erik T. [2001]. *Learning XML*. 1st Edition. O Reilly, 2001. ISBN 9780596000464 (cited on page 75).
- Roberts, Caroline [2007]. “Mixing modes of data collection in surveys: A methodological review” (2007) (cited on page 12).
- Schnell, Rainer, Paul B Hill, and Elke Esser [2011]. *Methoden der empirischen Sozialforschung*. 9th Edition. Oldenbourg Verlag, 2011. ISBN 3486591064 (cited on pages 7–9).
- SF-36 Survey* [2013]. Sept. 17, 2013. <https://itunes.apple.com/de/app/sf-36-survey/id519060213?mt=8> (cited on page 24).

- Shore, James and Shane Warden [2009]. *The Art of Agile Development*. 1st Edition. O'Reilly Media, Inc., 2009. ISBN 9780596527679 (cited on pages 63–65).
- Survey Master* [2013]. Sept. 17, 2013. <https://itunes.apple.com/us/app/survey-master/id556544556?mt=8> (cited on page 24).
- SurveyMonkey* [2013]. Sept. 28, 2013. <https://de.surveymonkey.com> (cited on page 28).
- Surveyor* [2013]. Sept. 17, 2013. <http://touchmetric.com> (cited on page 27).
- TabSurvey* [2013]. Sept. 17, 2013. <http://www.tabsurvey.com/tabsurvey/site/home.html> (cited on page 21).
- TestFlight* [2013]. Feb. 4, 2013. <https://testflightapp.com> (cited on page 66).
- The Comma Separated Value (CSV) File Format* [2013]. Sept. 12, 2013. <http://www.creativyst.com/Doc/Articles/CSV/CSV01.htm> (cited on page 21).
- The Waterfall Model* [2013]. Oct. 12, 2013. <http://www.whatsupnew.com/the-waterfall-model/> (cited on page 64).
- TPKeyboardAvoiding* [2013]. July 22, 2013. <https://github.com/michaeltyson/TPKeyboardAvoiding> (cited on page 75).
- Vaughan-Nichols, S.J. [2009-02]. “Will Mobile Computing’s Future Be Location, Location, Location?” *Computer* 42.2 (2009-02), pages 14–17. ISSN 0018-9162. doi:10.1109/MC.2009.65 (cited on page 4).
- Vonhoegen, Helmut [2007]. *Einstieg in XML*. 4th Edition. Galileo Press, 2007. ISBN 9783836210744 (cited on page 76).
- Watson, Peter D et al. [2001]. “Adolescents’ Perceptions of a Health Survey using Multimedia Computer-Assisted Self-Administered Interview”. *Australian and New Zealand journal of public health* 25.6 (2001), pages 520–524 (cited on page 17).
- Wilson, Matthew W. [2012]. “Location-Based Services, Conspicuous Mobility, and the Location-Aware Future”. *Geoforum* 43.6 (2012), pages 1266–1275. ISSN 0016-7185. doi:10.1016/j.geoforum.2012.03.014 (cited on page 5).
- WiseHunch* [2013]. Sept. 17, 2013. <http://wisehunch.com> (cited on page 27).
- Worldwide Devices Shipments by Operating System* [2013]. Oct. 13, 2013. <http://www.gartner.com/newsroom/id/2525515> (cited on page 43).
- Worldwide Smartphone Sales to End Users by Operating System in 2Q13* [2013]. Oct. 13, 2013. <http://www.gartner.com/newsroom/id/2573415> (cited on page 43).
- Zoomerang* [2013]. Sept. 28, 2013. <http://www.zoomerang.com> (cited on page 28).