ulm university universität
uulm

Universität Ulm | 89069 Ulm | Germany

Fakultät für Ingenieurwissenschaften und Informatik
Institut für Datenbanken und Informationssysteme

Technische Konzeption und Realisierung einer dynamisch generierten Anwendung für prozess-orientierte Fragebögen am Beispiel der mobilen Android Plattform

Diplomarbeit an der Universität Ulm

Vorgelegt von:
ALEXANDER REISSER
alexander.reisesr@uni-ulm.de

Gutachter:
Prof. Dr. Manfred Reichert
Dr. Verena Künzle

Betreuer:
M. Sc. Johannes Schobel

2014

Fassung 19. Februar 2014

Satz: PDF-LaTeX

Kurzfassung

Fragebögen in Papierform sind in vielen Bereichen beispielsweise der Psychologie oder Medizin ein gängiges Handwerkszeug zur Erhebung von Daten. In der heutigen hochtechnologisierten Welt ist diese Methode jedoch nicht mehr zeitgemäß und zum großen Teil mit viel Aufwand verbunden. Aus diesem Grund wurden am Institut für Datenbanken und Informationssysteme der Universität Ulm bereits erste elektronische Fragebogen auf mobilen Endgeräten realisiert. Jedoch weisen diese Systeme einige verbesserungswürdige Aspekte auf, wie beispielsweise fest im Programmcode hinterlegte Fragebögen. Dadurch ist es nicht ohne weiteres möglich, kurzfristig auf Änderungen oder Erweiterungen zu reagieren. Hierzu ist der direkte Eingriff in den Programmcode notwendig, was einen hohen zeitlichen Aufwand bedeutet und nur von entsprechenden Experten durchgeführt werden kann.

Um die Nachteile der bisherigen Anwendungen zu kompensieren, wurde ein Projekt geschaffen, um ein neues Fragebogensystem zu realisieren. Das neue System gliedert sich in die drei Teilbereiche der Modellierung von Fragebögen, der zentralen Verwaltung und der Ausführung auf mobilen Endgeräten.

Gegenstand der hier vorgelegten Ausarbeitung ist die Realisierung einer Anwendung zur Durchführung von Umfragen am Beispiel der mobilen Android Plattform. Hierbei werden die Anforderungen, das Konzept und die Anwendung selbst vorgestellt. Als Hauptaspekte sind hier die dynamische Generierung der Benutzeroberfläche und der Einsatz einer Prozess Engine zur Steuerung der Fragebögen zu nennen. Zudem nehmen Anforderungen an die Usability eine wichtige Rolle ein. Darüber hinaus werden Kernpunkte der Implementierung im Detail diskutiert. Mithilfe des Ausblicks und den alternativen Anwendungsmöglichkeiten werden aktuelle und zukünftige Polyvalenzen der mobilen Anwendung aufgezeigt.

Danksagung

Als ich vor gut einem halben Jahr auf der Suche nach einem Thema für die Diplomarbeit war, wurde ich am Institut für Datenbanken und Informationssysteme der Universität Ulm sehr herzlich aufgenommen. Deshalb gilt mein bester Dank der Leitung dieses Instituts, Herrn Prof. Dr. Reichert. Er versteht es zudem, in den Vorlesungen nicht nur Wissen zu vermitteln, sondern auch Interesse für das Tätigkeitsgebiet zu wecken. Ebenfalls möchte ich mich für seine Tätigkeit als Erstgutachter bedanken.

Ein ganz besonderer Dank gilt meinem Betreuer Johannes Schobel. Er hat sich überdurchschnittlich viel Zeit für unzählige Diskussionen genommen und hatte immer ein offenes Ohr für meine Probleme. Des Weiteren möchte ich mich für die grammatikalische und sprachliche Überprüfung meiner Arbeit bei Johannes bedanken. Es ist schon erstaunlich, zu welchen Uhrzeiten man korrigierte Fassungen von ihm erhält. Aus diesen Gründen kann ich Johannes mit bestem Gewissen als Betreuer empfehlen.

Den größten Dank möchte ich meiner Freundin Antje aussprechen. Sie hat sich während der Realisierung bestens um mich gekümmert und mich in allen Phasen motiviert. In der „heißen Phase“ erhielt ich von ihr stets psychisch und kulinarisch Unterstützung.

Ein weiterer Dank gilt dem restlichen Team des Projekts. Danke Arnim Schindler, Juri Schulte und Karoline Blendinger für die gute Zusammenarbeit und Hilfestellungen. Ein weiterer Dank gilt Steffen Scherle für das grandiose Oberflächendesign. Besonders bedanken möchte ich mich bei Fabian Maier für die Realisierung und Betreuung der Prozess Engine. Wir benötigten doch einige Iterationen, bis die benötigten Funktionen erarbeitet wurden.

Meinen Arbeitskollegen danke ich an dieser Stelle für die indirekte Unterstützung. Sie hielten mir währen den stressigen Phasen den Rücken frei.

Nicht zuletzt gehört ein ganz besonderer Dank meiner Familie, allen voran meinen Eltern Gottfried und Gertrud, so wie meiner Schwester Carolin. Ohne ihren Rückhalt und deren Unterstützung wäre dieses Studium nicht möglich gewesen.

Inhaltsverzeichnis

1

Einleitung

In der heutigen modernen Kommunikationswelt sind viele Wege und Methoden vorhanden, um Daten und Informationen zu gewinnen. Eine solche Methode ist die Erhebung mit Hilfe eines Fragebogens. Diese Art der Informationsgewinnung ist sehr weit verbreitet, sowie mit vielen unterschiedlichen Erscheinungsformen einhergehend [Sch12]. So gibt es beispielsweise Fragebögen, die von Besuchern öffentlicher Einrichtungen ausgefüllt werden, um den internen organisatorischen Ablauf zu analysieren und zu verbessern. Ein weiteres Anwendungsbeispiel ist die Mikrozensus Umfrage, welche eine statistische Erhebung ist, die dazu dient, die im Rahmen von Volkszählungen erhobenen Daten in kurzen Zeitabständen zu überprüfen und gegebenenfalls zu korrigieren. Der klassische Fragebogen ist meist in Papierform und wird von Hand ausgefüllt. Im Zuge der Akzeptanz des Internets entstehen vermehrt Fragebögen im Online-Format. Diese können elektronisch ausgefüllt und direkt versendet werden. Gegenüber der klassischen papierbasierten Variante, bieten elektronische Fragebögen diverse Vorteile. So liegen die

erhobenen Daten beispielsweise direkt in digitaler Form für die anschließende computergestützte Analyse bereit. Desweitern wird Papier gespart, was aus ökologischer und finanzieller Sicht gerade bei großen Umfragen und Erhebungen nicht zu vernachlässigen ist.

Im Zuge der rasanten Verbreitung von mobilen Endgeräten (beispielsweise Smartphones oder Tablet PCs) bietet es sich an, elektronische Umfragen mit solchen Geräten durchzuführen. Dazu wurden im Rahmen von Abschlussarbeiten bereits Anwendungen realisiert, welche Fragebögen in elektronischer Form darstellen [Sch12], [Mai12]. Jedoch sind die Fragebögen dieser mobilen Anwendungen themenspezifisch und weisen einige Nachteile auf (siehe Kapitel 2). Änderungen oder Erweiterungen an diesen Anwendungen sind nur schwer umsetzbar, und erfordern einen hohen Programmieraufwand. Zudem ist eine komplexe Ablaufsteuerung der Fragebögen nur schwer möglich, da diese in der Programmlogik enthalten ist.

Ein neuer Ansatz, um den Inhalt und die Ablauflogik eines Fragebogens auch nachträglich einfach ändern zu können, ist die Trennung von Fragebogen und Anwendung. Dadurch sind Fragebogeninhalt und Ausführungslogik nicht mehr direkt in der Anwendung enthalten und können separat in einer Konfigurationsumgebung erstellt werden. Dieser so erstellte Fragebogen, welcher eine Seitenbeschreibung und die dazugehörige Ausführungslogik enthält, wird anschließend auf das mobile Endgerät geladen. Dort generiert die Anwendung dynamisch anhand der Seitenbeschreibung eine Benutzeroberfläche. Die auf dem mobilen Endgerät enthaltene Prozess Engine interpretiert dabei die Ausführungslogik und steuert so den Fragebogenablauf. Dieser Ansatz verspricht einige Vorteile:

- Auf Änderungen der Ablauflogik von Fragebögen kann schnell reagiert werden. Nur der modellierte Prozess wird dabei angepasst, es ist somit kein expliziter Programmieraufwand erforderlich.
- Die Anwendung auf dem mobilen Endgerät muss dabei nicht erneuert werden, da die Benutzeroberfläche des Fragebogens von der Anwendung dynamisch generiert wird. Lediglich der Fragebogen selber wird dabei ausgetauscht, welcher dabei als Informationsträger dient.

- Durch Benutzung von Interaktionselementen für die Darstellung der Fragen in neutral gehaltenem Design, können die Fragebögen in unterschiedlichen Domänen verwendet werden.
- Die Mobilität ermöglicht das Durchführen von Interviews an beliebigen Orten unter Verwendung von mobilen Endgeräten.

Dieser neuartige Ansatz wird am Institut für Datenbanken und Informationssysteme der Universität Ulm entwickelt und erforscht. Das System setzt dabei auf Prozess-Technologie, um Fragebögen zu erstellen und anschließend auch auszuführen. Gegenstand der hier vorliegenden Arbeit ist die Realisierung einer dynamisch generierten Anwendung für mobile Endgeräte zur Durchführung von prozess-orientierten Fragebögen.

1.1. Hintergrund zum Gesamtsystem

Wie bereits in Kapitel 1 erläutert, ist die hier dargelegte Arbeit ein Teilbereich des Gesamtsystems QuestionSys. Die Abbildung 1.1 zeigt einen schematischen Überblick über die unterschiedlichen Komponenten des Systems.

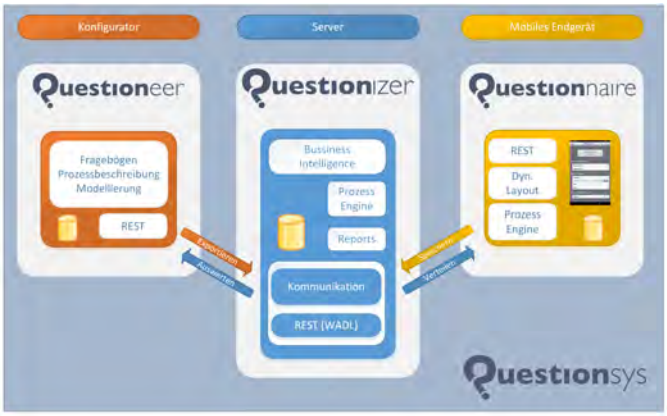


Abbildung 1.1.: Überblick des Gesamtsystems

Konfigurator: Unter Verwendung des Konfigurators werden Fragebögen und deren Ablauflogik modelliert. Somit ist der Fragebogenaufbau nicht mehr fest in der mobilen Anwendung codiert, sondern wird getrennt erstellt. Gespeichert werden alle modellierten Daten auf dem zentralen Server. Die Konzeption und das Design der Konfiguratorumgebung wurde dabei in einer separaten Abschlussarbeit entwicklet [Sch14].

Server: Der zentrale Server speichert alle Fragebögen und die dazu erhobenen Antworten. Die Antworten werden von den mobilen Endgeräten nach der Befragung zum Server gesendet. Zusätzlich dient der Server als Kommunikationsplattform, welche von Konfigurator und dem mobilen Endgerät genutzt werden. Die Schnittstellen sind dabei als Webservice realisiert. Des Weiteren ist für den Server eine Möglichkeit zur Analyse der Antworten angedacht.

Mobiles Endgerät: Eine generische Anwendung, welche auf dem mobilen Endgerät installiert wird, stellt dem Benutzer die auf dem Konfigurator modellierten Fragebögen dar. Dazu werden über die Service-Schnittstelle am Server die Fragebögen angefragt und auf dem mobilen Endgerät persistiert. Das Speichern selber wird von der, auf dem mobilen Endgerät enthaltenen, leichtgewichtigen Prozess Engine, durchgeführt. Über diese wird auch der dynamische Aufbau der Benutzeroberfläche gesteuert, da sie die modellierte Ausführungslogik des Fragebogens interpretiert. Die vom Benutzer erhobenen Antworten werden direkt an die Prozess Engine geleitet und gespeichert, damit keine Daten verloren gehen. Nach Abschluss eines Fragebogens werden die erhobenen Daten an den zentralen Server gesendet.

1.2. Aufbau der Arbeit

Der weitere Verlauf der Arbeit gliedert sich in Kapitel 2, welches Forschungsarbeiten und auf dem Markt zugängliche Anwendungen mit ähnlichem Fokus diskutiert. Anschließend werden Nachteile aus diesen diskutierten Anwendungen extrahiert und in Bezug auf die zu realisierende Anwendung gebracht. Die wichtigsten Anforderungen an die zu realisierende Anwendung werden in Kapitel 3 aufgeführt. Dabei wird zwischen funk-

tionalen und nichtfunktionalen Aspekten unterschieden. Im Kapitel 4 wird der Aufbau der Anwendung schematisch betrachtet. Dabei wird besonders auf die Aspekte Systemarchitektur, Aufbau des Fragebogens, Prozess Engine und dem Zusammenspiel der Komponenten eingegangen. Eine Einführung in das User Interface der Anwendung wird in Kapitel 5 gegeben. Dazu wird der Ablauf der Anwendung anhand von Screenshots nachgestellt. Kapitel 6 bietet einen tieferen Einstieg in die Realisierung auf Basis des Android Frameworks. Das Kapitel 7 fasst die Arbeit zusammen und gibt einen Ausblick auf nachträgliche Erweiterungen.

2

Verwandte Arbeiten

Dieses Kapitel diskutiert Forschungsarbeiten mit ähnlichem Fokus und stellt diese einander gegenüber. Dabei sollen bestehende Nachteile dieser Arbeiten extrahiert und in Bezug mit der hier vorliegenden Arbeit gebracht werden.

2.1. Masterarbeit von Maximilian Schmid

Mit der Arbeit *"Technische Konzeption und Realisierung der Anwenderumgebung für ein generisches Fragebogensystem zur IT-gestützten Durchführung von evaluierten Studien der Klinischen Psychologie"* [Sch12] wurde ein webbasierter Fragebogen-Client erstellt (siehe Abbildung 2.1).

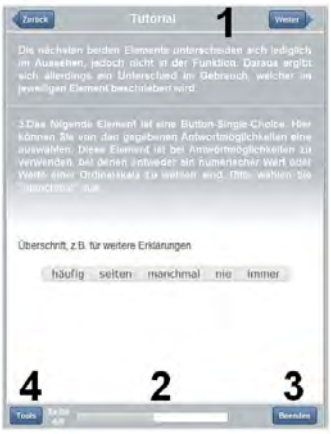


Abbildung 2.1.: Exemplarische Seite eines Fragebogens aus der diskutierten Arbeit [Sch12]

Durch die Verwendung der Web-Technologie ist die Client Anwendung plattformunabhängig und kann auf verschiedensten Endgeräten eingesetzt werden. Im Speziellen wird hier die Technik der WebApp genutzt, wodurch mobile Webseiten unter Verwendung des PhoneGap Frameworks in native WebApps für die jeweiligen Betriebssysteme konvertiert werden. Dadurch ist es möglich auf Ressourcen des Betriebssystems zuzugreifen, was aus dem Browser nur eingeschränkt möglich ist. Somit kann eine Umfrage auf beliebigen mobilen Endgeräten durchgeführt werden.

Positiv hervorzuheben ist die Trennung von Fragebogen und der eigentlichen Anwendung. Die Anwendung auf dem mobilen Endgerät erstellt dabei für den Benutzer eine Bedienoberfläche mit welcher die Umfrage durchgeführt werden kann. Hierbei werden sehr domänenspezifische Interaktionselemente verwendet, da die Anwendung im Bereich der Medizin eingesetzt wird. In der hier vorliegenden Arbeit werden ausschließlich Interaktionselemente in neutralem Design verwendet, um Umfragen in unterschiedlichen Bereichen durchführen zu können.

Die Reihenfolge, in welcher die Fragebogenseiten dem Benutzer dargestellt werden sollen, kann in der diskutierten Arbeit [Sch12] vorgegeben werden, jedoch ist es nicht möglich komplexe Abläufe zu definieren. Aus diesem Grund ist in der hier vorliegenden Arbeit die Steuerung der Fragebogenseiten durch eine leichtgewichtige Prozess Engine

vorgesehen. Die Prozess Engine wurde eigens für diesen Anwendungsfall entwickelt und befindet sich auf dem mobilen Endgerät. Dadurch wird die Interpretation von komplexer Ablauflogik für Fragebögen unterstützt und fordert keine Änderung des Programmcodes.

Des weiteren ist im Fragebogen-Client ein Offline-Betrieb vorgesehen. Dazu werden die auszuführenden Fragebögen auf dem mobilen Endgerät gespeichert. Die erhobenen Antworten vom Benutzer persistiert die Anwendung dabei ebenfalls lokal. Nachteilig Anzumerken ist hierbei allerdings, dass die erhobenen Daten im Offline-Betrieb vom Benutzer gelöscht werden können, ohne dass diese zuvor auf einen zentralen Server persistiert wurden. Das Konzept der hier dargelegten Arbeit sieht hierfür eine Datenbank vor. Durch die Prozess Engine werden alle erhobenen Daten während einer Umfrage in der lokalen Datenbank persistiert. Diese können im Passwort geschützten administrativen Bereich der Anwendung erst nach erfolgreicher Synchronisierung mit dem Server gelöscht werden.

Ein weiterer wichtiger Aspekt ist die Mehrsprachigkeit der Fragebögen, da Befragungen oft an verschiedenen Orten und in unterschiedlichen Ländern durchgeführt werden. In der hier diskutierten Arbeit wird dies zwar unterstützt, jedoch muss ein Fragebogen mehrmals in den jeweiligen Sprachen erstellt und auf das Endgerät geladen werden. In der hier vorliegenden Arbeit kann die Sprache für einen Fragebogen vor dem Start umgestellt werden. Die Sprache der Anwendung und die des Fragebogens werden dabei getrennt voneinander behandelt. Detailliert wird dieser Aspekt in Kapitel 3.2.1 diskutiert.

Nicht zu vernachlässigen ist der Aspekt der unterstützen Fragetypen. In der Arbeit [Sch12] sind sechs verschiedene Typen von Fragen umgesetzt. Dies sind neben der Button Single- und Button Multiple Choice, das Äquivalent in Tabellenform (Table Single Choice bzw. Table Multiple Choice), sowie der Slider und die Freitextantwort. Auffallend ist hierbei, dass es beispielsweise beim Slider keinen neutralen Zustand gibt. Somit ist initial bereits ein Wert ausgewählt, welcher aus psychologischer Sicht den Benutzer in der Entscheidungsphase beeinflusst. Dies kann beispielsweise bei einem Zweistellungsschalter der Fall sein. In der realisierten Anwendung wurde deshalb für jedes Interaktionselement ein initialer neutraler Status implementiert. Im vorherigen Beispiel wäre das eine zusätzliche temporäre Stellung in der Mitte des Schalters.

Zusammenfassend ist zu erwähnen, dass der realisierte Fragebogen-Client aus der Arbeit [Sch12] einen sehr dynamischen Aufbau in Bezug auf die Darstellung der Fragebögen aufzeigt, jedoch ebenso mit einigen Nachteilen behaftet ist. Dies ist beispielsweise die Tatsache, dass Antworten im Offline-Betrieb vom Benutzer gelöscht werden können, oder einige Interaktionselement keinen neutralen Zustand aufweisen. Positive Aspekte sind die Plattformunabhängigkeit mittels PhoneGap, sowie die Trennung des Fragebogeninhalts von der Anwendung.

2.2. Bachelorarbeit von Fabian Maier

In der Arbeit *„Entwicklung eines mobilen und Service getriebenen Workflow-Clients zur Unterstützung von evaluierten Studien der klinischen Psychologie am Beispiel der AristaFlow BPM Suite und Android“* [Mai12] wurde eine Anwendung für mobile Endgeräte entwickelt. Als Voraussetzung für den Einsatz dieser Anwendung wird eine serverseitige Prozess Engine definiert, mit welcher der Fragebogenablauf gesteuert wird. Diese beiden Komponenten kommunizieren dabei über einen vom Server bereit gestellten Web Service.

In der hier diskutierten Arbeit ist positiv zu erwähnen, dass der Fragebogen getrennt von der Anwendung auf den mobilen Geräten betrachtet wird. Somit sind auch hier Änderungen im Aufbau und Ablauf des Fragebogens einfach zu realisieren. Die Anwendung selber muss dabei nicht abgeändert werden. Die vorausgesetzte Prozess Engine, welche auf einem externen Server arbeitet, kann im ersten Moment als Vorteil gesehen werden, denn dadurch ist das interpretieren von komplexer Ablauflogik für Fragebögen möglich. Auf den zweiten Blick jedoch bringt sie auch Nachteile mit sich, denn für die Durchführung einer Umfrage wird eine dauerhafte Verbindung mit dem Server vorausgesetzt. Somit kann die Client-Anwendung mit hoher Wahrscheinlichkeit in Dritte-Welt-Ländern nicht eingesetzt werden, da dort kein flächendeckendes mobiles Internet verfügbar ist. Ebenso ist das Verhalten der Anwendung bei sporadischen Verbindungsabbrüchen zum Server nicht erwartungskonform. Falls die Verbindung verloren geht, ist die Anwendung nicht mehr bedienbar, da sie auf eine Wiederanbindung wartet.

In seltenen Fällen kann es sogar vorkommen, dass ein manueller Eingriff auf dem Server notwendig ist, um die Umfrage nach einem Verbindungsabbruch fortzuführen. In der hier vorliegenden Arbeit wird keine dauerhafte Verbindung zu einem Server vorausgesetzt, da sich die Fragebögen und die eingesetzte Prozess Engine direkt auf dem mobilen Endgerät befinden. Dadurch sind alle, für den Offline-Betrieb benötigten Ressourcen direkt auf dem Endgerät vorhanden. Ebenso minimieren sich die Übertragungszeiten bei der Kommunikation mit der Prozess Engine, da lokal kommuniziert werden kann.

Im Vergleich mit der im Kapitel 2.1 diskutierten Arbeit ist die Plattformunabhängigkeit nicht gegeben, da die Anwendung nativ für das Android Betriebssystem implementiert wurde. Um die auf dem Server entworfenen Fragebögen mit mobilen Endgeräten anderer Betriebssysteme (beispielsweise iOS oder Windows) benutzen zu können, müssen dafür separate Anwendungen realisiert werden. Ein weiterer positiver Aspekt, welcher durch die Verwendung einer Prozess Engine herbeigeführt wird, ist die Möglichkeit, eine Umfrage zu unterbrechen und zu einem späteren Zeitpunkt fortzusetzen. Dieser Aspekt ist direkt in das Konzept der hier vorgelegten Arbeit eingeflossen.

Wie bereits beim Betrachten der hier diskutierten Arbeit festgestellt wurde, sind auch für die meisten Interaktionselementen zum Beantworten von Fragen keine neutralen Werte berücksichtigt worden. Somit ist beispielsweise bei einem Slider stets ein konkreter Wert vorausgewählt. In der hier vorgelegten Arbeit wird dieses Problem durch Verwendung eigens realisierter Interaktionselemente behoben. Diese weisen alle eine neutrale Position auf, welche initial gesetzt wird. Bei einigen dieser Elemente ist es auch nachträglich möglich den neutralen Wert auszuwählen, um sich beispielsweise einer Anwort zu enthalten.

Wie bereits im Kapitel 2.1 aufgeführt, sind die Interaktionselemente der hier diskutierten Arbeit ebfalls für den Bereich Medizin ausgerichtet. Somit können nur domänenspezifische Umfragen mit der Client-Anwendung durchgeführt werden. In der hier vorgelegten Arbeit werden ausschließlich Interaktionselemente mit neutralem Design eingesetzt, welche eigens für diesen Zweck realisiert wurden. Damit sind mit der neu realisierten Anwendung, Umfragen in verschiedensten Bereichen möglich.

Zusammengefasst lässt sich sagen, dass durch den Einsatz einer Prozess Engine auf dem Server komplexe Fragebogenabläufe einfach abgebildet werden können, jedoch Probleme bei Verbindungsabbrüchen auftreten. Ohne Verbindung zum Server ist die Durchführung einer Umfrage nicht möglich. Weiterhin ist das Konzept der Trennung des Fragebogens von der Anwendung positiv hervorzuheben, wobei sich die Modellierung der Fragebögen mit dem dafür vorgesehenen Editor eher mühsam gestaltet.

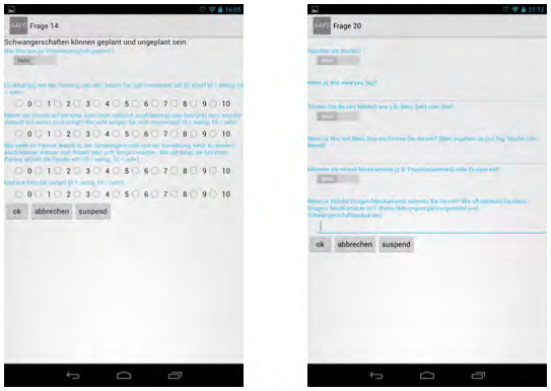


Abbildung 2.2.: Exemplarische Fragebogenseiten der hier diskutierten Arbeit [Mai12]

2.3. QuickTabSurvey

Das kommerzielle Produkt QuickTabSurvey [Qui14] ist laut eigenen Angaben des Unternehmens in Bezug auf die Datenerfassung, Marktführer im Bereich der Marktforschung. QuickTabSurvey ist eine frei konfigurierbare Datenerfassungsplattform, welche beispielsweise für Kundenbefragungen verwendet werden kann.

Ein neuer Fragebogen kann dabei sehr einfach über eine online Administrationsseite erstellt werden. Dazu stehen mehr als 35 verschiedene Typen von Fragen zur Auswahl, mit welchen beispielsweise ein Text, oder ein boolescher Wert erfasst werden kann (siehe Abbildung 2.3). Die Integration von Medien (beispielsweise Bilder und Videos) ist ebenso möglich. Die Erweiterung der bestehenden Fragentypen um eigens entworfene Typen wird jedoch nicht unterstützt, was als Nachteil zu werten ist. In der hier vorgelegten Arbeit ist die Architektur der Anwendung so konzipiert, dass neue Frage-Typen leicht

integriert werden können. Somit wird ein maximum an Flexibilität und Erweiterbarkeit erreicht.

Bei der Gestaltung der Fragebögen für ein QuickTabSurvey ist es dem Benutzer überlassen ober er alle Fragen auf einer Seite darstellt, oder diese auf mehrere Seiten verteilt. Falls mehrere Seiten für einen Fragebogen existieren, kann zusätzlich eine einfache Logik zur Ausführungsreihenfolge definiert werden. Hierbei sind nur einfache Verzweigungen, beispielsweise anhand des eingegebenen Geschlechts, möglich. Um diesen Aspekt unterscheidet sich die hier vorgelegte Arbeit stark vom kommerziellen Produkt, da auf den mobilen Endgeräten zu Steuerung der Fragebögenseiten eine Prozess Engine verwendet wird. Mit dieser ist es möglich komplexe Abläufe bei der Ausführung von Fragebögen zu realisieren.

Die Ausführung einer Umfrage findet auf mobilen Endgerät für die Plattformen iOS und Android statt. Mit der mobilen Anwendung kann der vorher entworfene Fragebogen auf das mobile Endgerät geladen und gestartet werden. Zur Durchführung von Umfragen wird keine Verbindung mit dem Internet vorausgesetzt. Diese wird nur benötigt, um Fragebögen auf das Endgerät zu laden und die erhobenen Daten zurück an die zentrale Datenerfassungsplattform zu senden. Falls während der Erfassung von Daten eine Verbindung zum Server besteht, werden diese direkt zum Server gesendet und gespeichert. Auf der Administrationsseite besteht die Möglichkeit, die empfangenen Daten live zu analysieren und zu exportieren (beispielsweise als PDF oder als Rohdaten zur weiteren Verarbeitung durch eigene Anwendungen).

Zusammenfassend lässt sich über QuickTabSurvey sagen, dass es bei der Erstellung, als auch bei der Visualisierung von Fragebögen sehr dynamische Ansätze aufweist und leicht zu Verwenden ist. Der Ablauf des Fragebogens kann dabei nur begrenzt gesteuert werden, was sich als Nachteil erweist. Zusätzlich bietet es auf der Administrationsseite Analyse- und Exportmöglichkeiten. Die anfallenden Kosten für die Nutzung des Systems können hierbei als Nachteil aufgeführt werden. Ebenso der Aspekt, dass die Fragebogentypen nicht selbst erweitert werden können.

(a) Auswahl eines Kartenpunkts (b) Auswahl des Geschlechts

Abbildung 2.3.: Fragen Elemente von QuickTabSurvey [Qui14]

2.4. Fazit

In der hier vorliegenden Arbeit sollen, sofern möglich, die Vorteile der bisherigen Arbeiten übernommen und die Nachteile beseitigt werden. Zum besseren Verständnis werden die verschiedenen Kriterien in der Tabelle 2.1 gegenüber gestellt.

Kriterien	Schmid	Maier	QuickTabSurvey	Questinnaire
Offline Verfügbar	ja	nein	ja	ja
Process Engine	nein	Extern	nein	ja
Plattformunabhängig	ja	nein	nein	nein
Plattform	Web	Android	iOS/Android	Android
Zentrale Verwaltung	ja	ja	ja	ja
Editierbarkeit	ja	ja	ja	ja
Erweiterbarkeit	ja	ja	nein	ja
Neutrales Design	nein	nein	ja	ja
Neutraler Zustand (Bedienelemente)	nein	nein	nein	ja

Tabelle 2.1.: Gegenüberstellung Arbeiten

Ein beseitigter Nachteil in Bezug auf die diskutierte Arbeit [Sch12] ist die schlechte Unterstützung in der Ablaufsteuerung von Fragebögen. Dazu wurde eine eigens für diesen Anwendungsfall konzipierte Prozess Engine realisiert, welche sich lokal auf dem mobilen Endgerät befindet. Dadurch ist die Steuerung von Fragebogenseiten unter Verwendung von komplexen Ablaufmustern möglich.

Ebenso ist in der Anwendung ein Offline-Betrieb realisiert, wodurch Umfragen ohne Verbindung zu einem Server durchgeführt werden können. Dies ist nur deshalb möglich, da sich die eingesetzte Prozess Engine lokal auf dem mobilen Endgerät befindet und die relevanten Ressourcen wie beispielsweise Fragebögen direkt auf dem Endgerät gespeichert sind. In der diskutierten Arbeit [Mai12] war dies nicht möglich, da zur Ausführung eine dauerhafte Verbindung zum Server vorausgesetzt wurde.

In Bezug auf die Erweiterbarkeit grenzt sich die hier dargelegte Arbeit klar von der kommerziellen Anwendung QuickTabSurvey ab. Dort ist es nicht vorgesehen, den Katalog von Fragen-Typen um eigene Typen zu erweitern. In der neu realisierten Anwendung ist die Integration zusätzlicher Typen von Beginn an konzeptuell berücksichtigt und implementiert worden.

Eine weitere Abgrenzung zu den bisher diskutierten Anwendungen ist die Gestaltung der Interaktionelemente. Diese sind in der hier vorgelegten Arbeit neutral gehalten und können dadurch in verschiedensten Domänen eingesetzt werden. Desweiteren wurde auf den psychologischen Aspekt von bereits vorausgewählten Bedienelementen geachtet, welcher in allen bisher betrachteten verwandten Arbeiten zu finden war. Bei Zweistellungsschaltern ist dort inital bereits eine mögliche Antwort vor ausgewählt. Dieses Problem wurde in der neu realisierten Anwendung durch Verwendung eines neutralen Zustandes gelöst, welcher initial ausgewählt ist.

3

Anforderungen

In diesem Kapitel werden die wichtigsten Anforderungen aufgeführt und erläutert, welche als Grundlage für die Realisierung des generischen Fragebogensystems auf der mobilen Android Plattform verwendet wurden. Dabei spielen viele funktionale Aspekte eine Rolle, doch auch eine grosse Anzahl an nichtfunktionalen Anforderungen dürfen dabei nicht außer Acht gelassen werden.

Auf die Anfertigung eines Lasten- / Pflichtenheftes wurde bei diesem Projekt verzichtet, da anfangs die Anforderungen an das System noch nicht exakt sichtbar waren. Viel mehr wurde hier die Methodik der agilen Softwareentwicklung eingesetzt. Eine komplette theoretische Planung des Gesamtsystems (Questioneer, Questionizer, Questionnaire) inkl. der Umsetzung auf den verschiedenen Plattformen wäre in der begrenzten Zeit nicht möglich gewesen.

Das allgemeine Ziel dieser hier vorliegenden Arbeit ist es, einen Fragebogen Client zu erstellen, welcher ein User Interface zum Beantworten der Fragen basierend auf Basis einer Beschreibungssprache generiert. Somit ist die Oberflächengestaltung nicht an eine spezifische Domäne gebunden und muss deshalb neutral, transparent, einfach und verständlich gehalten werden, damit sich Benutzer unterschiedlichen Bildungsgrades und Anwendungsdomänen mit der Software auseinander setzen können.
Im Folgenden sind die aufgeführten Oberflächenelemente stark an die Psychologie bzw. Medizin angelehnt, da aus bereits realisierten Projekten sehr detailiertes Hintergrundwissen vorhanden ist.

3.1. Nichtfunktionale Anforderungen

Die nichtfunktionalen Anforderungen beschreiben wie gut ein System etwas leisten soll (qualitativ) und stehen nicht direkt in Interaktion mit der Funktion des eigentlichen Systems.

3.1.1. Usability

Die *Usability* (auch Gebrauchstauglichkeit genannt) bezeichnet den Grad, in welchem ein Produkt oder System durch bestimmte Benutzer verwendet werden, um bestimmte Ziele (Effektivität, Effizienz, Zufriedenheit) im Nutzungskontext zu erreichen. Unterschieden wird dabei zwischen zwei Kontexte [Off12]:

- **Ease of Use**: Sehr gut ausgebildete oder geschulte Benutzer, welche häufig mit dem System in Berührung kommen. Die Dialoge können hier komplexer sein.
- **Ease of Learning**: Benutzer ohne Schulung, welche das System nur sporadisch nutzen. Das System muss selbsterklärend sein, damit sich solche Benutzer einfach zurechtfinden, und ihre Aufgaben erledigen können.

Dieser Aspekt des Nutzerkontextes nimmt einen besonders großen Stellenwert ein, da für das Questionnaire Projekt drei unterschiedliche Benutzergruppen in Betracht gezogen werden müssen.

So gibt es die Gruppe der *Administratoren*, von denen auszugehen ist, dass sie ausreichend in der Bedienung der Software geschult sind. Somit ist es dieser Gruppe ohne größere Probleme möglich, administrativen Aufgaben, wie das Einspielen neuer Fragebögen, Löschen von Fragebögen, Zurücksenden von ausgefüllten Fragbögen und Verwalten von Benutzern zu übernehmen. Für diese Tätigkeiten muss kein besonderes Augenmerk auf die effiziente Gestaltung der Benutzeroberfläche gelegt werden, da für geschulte Benutzer die Oberfläche nicht zwingend ergonimisch aufgebaut sein muss.

Als zweite Benutzergruppe wurden die *Interviewer* identifiziert, welche die Befragungen durchführen. Diese können vor Erhalt des mobilen Endgeräts ebenfalls eine Einweisung erhalten, um sich auf dem Gerät zurecht zu finden. Da davon auszugehen ist, dass dieser Personenkreis keine Expertenbenutzer sind und nicht regelmäßig Daten mittels der elektronischen Fragebögen erheben, muss die Benutzeroberfläche der Anwendung einfach und selbsterklärend gestaltet sein.

Die *Befragten* selbst bilden letztendlich die dritte Gruppe der zu betrachtenden Personen. Bei einzelnen Fragen oder ganzen Fragebögen ist es vorgesehen, dass die Probanden selbst tätig werden müssen und die Fragen autonom lesen, sowie die dazu gehörigen Antworten auf dem mobilen Endgerät eingeben. Dadurch ist es möglich die Antworten mit weitere Daten anzureichern (wie beispielsweise das Antwortverhalten, die Antwortzeit, Bilder oder Bewegungsmuster). Um dieses zu bewerkstelligen muss die Bedienung für die Laien-Benutzer sehr intuitiv, transparent und selbsterklärend sein, da eine anfängliche Schulung aus zeitlichen Gründen ausgeschlossen ist. Zu Erreichen ist das durch eine Reduktion der Funktionalität, sinnvolle Gruppierung der Funktionalität, sinnvolle Anordnung und effiziente Navigation zwischen Dialogmasken.

Diese Aspekte wurden bei der Umsetzung berücksichtigt, wodurch einige *neue Interaktionselemente* geschaffen werden mussten, um die Einfachheit zu gewährleisten.

3.1.2. Performance

Unter Performance versteht man die Art und Weise, wie ein System auf die Handlungen eines Anwenders reagiert, wieviel Zeit es dazu benötigt und wie der Benutzer die Reaktion wahrnehmen und interpretieren kann.

Dieser Aspekt, welcher auch als Gütekriterium bei Usability Tests eingesetzt wird, spielt in diesem Projekt eine nicht zu unterschätzende Rolle, da jede Anwendung mit Benutzerinteraktion, auf ihr Laufzeitverhalten geprüft werden muss.

Im Einzelnen sind folgende Aspekte näher zu betrachten:

- **Reaktionszeit der Anwendung**: Die Reaktionszeit der Anwendung wird in der Fachliteratur [Usa99] im Durchschnitt auf einen Wert kleiner einer Sekunde gesetzt. Sollte dieser Wert nicht eingehalten werden können, so besteht die Möglichkeit dem Benutzer eine Meldung anzuzeigen, welche ihm visualisiert, dass das System weiterhin arbeitet, jedoch noch einige Zeit für die Ausführung der durch den Benutzer angeforderten Aufgabe benötigt. Wird dem Anwender keine solche Rückmeldung signalisiert, kann dieser nicht unterscheiden werden, ob das System weiterhin arbeitet oder die Ausführung der Funktion bereits abgebrochen wurde. Hierbei gilt es trotz alledem, die Geduld des Anwenders nicht zu sehr zu strapazieren, da die Akzeptanz der Software durch zu lange Wartezeiten sinkt.

- **Feedback der Anwendung**: Unter Feedback versteht man die Interaktion der Software mit dem Benutzer. Jeder Aktion des Anwenders muss vom System schnellstens ausgeführt und beantwortet (visualisiert) werden. Dies kann beispielsweise durch das Einfärben eines Buttons beim Drücken oder durch das Öffnen eines Dialoges geschehen. In der hier vorliegenden Arbeit soll jeder Button ein blaue Schattierung erhalten um den Zustand eindeutig interpretieren zu können. Darüber hinaus sollen sämtliche Fehlermeldungen vom System protokolliert und einem Administrator angezeigt werden. Im Fragebogen System wird dies durch allgemein gehaltene Dialoge für die Kategorien `Message` und `Error` umgesetzt.

3.1.3. Design

Für das Design dieser Anwendung wurde in einer separaten Abschlussarbeit ein Styleguide inklusive diverser Mockups im iOS Style erstellt [Sch14]. Unter Berücksichtigung von emotionalen und psychologischen Aspekten ist das vorgegebene Design auf der Android Plattform umzusetzen. Dazu müssen jedoch einige Elemente aus technischen und urheberrechtlichen Gründen an den Android Style angepasst werden.

Eine ausführliche Beschreibung der User Interface Elemente ist unter Kapitel 3.2.9 zu finden.

3.1.4. Plattformunabhängigkeit

Eine direkte Plattformunabhängigkeit ist hier nicht gegeben, da die Software direkt für die Android Plattform entwickelt wurde. In der zukünftigen Weiterentwicklung des Gesamtprojektes ist jedoch ein separater Client für die iOS Plattform angedacht.
Gründe für die Wahl der *nativen Anwendung* als Realisierungsform sind im Kapitel 4.1.3 beschrieben.

Bezüglich des Aspektes Portabilität ist allerdings noch zu erwähnen, dass es möglich sein soll, die Software auf diverse Endgeräte zu Installieren, welche Android in der Version größer oder gleich 4.4 als Betriebssystem aufweisen. [Kü12]

3.1.5. Erweiterbarkeit und Wartbarkeit

Der Aspekt *Erweiterbarkeit* ist in drei Punkte zu unterteilen.

- Als erstes muss es möglich sein, neue Sprachen für die Anwendung zu definieren. Hierzu bietet die Android Plattform bereits einfache Methoden dies zu bewerkstelligen. Dabei ist jede Sprache als separate Datei definiert, welche sämtliche Übersetzungen enthält. Somit können neue Sprachen im Anwendungskontext sehr einfach hinzugefügt werden.

- Zweitens besteht die Möglichkeit die Software um neue Elemente zur Interaktion für den dynamisch generierten Teil des Fragebogens zu erweitern. Wie bereits erwähnt wurde, ist das Layout und die Interaktionselemente in der hier vorliegenden Arbeit sehr an medizinische und psychologische Fragetypen angelehnt. Um allerdings beispielsweise eine Zufriedenheitsumfrage für Reisebüros zu erstellen, werden sicherlich noch weitere Fragen-Typen benötigt. Hierzu wäre sicherlich eine Summenfrage hilfreich. Beispiel: Verteilen Sie 100 Punkte auf folgende Antworten: Urlaub ist am schönsten: in den Bergen | am Meer | zuhause.
- Als Drittes darf die Erweiterung der Anwendung um zusätzliche neue Komponenten nicht vernachlässigt werden. Zukünftig können weitere Anforderungen aufkommen, welche in die bestehende Software integriert werden müssen. Hierzu gilt es den Code möglichst sauber und strukturiert zu gestalten. Eine neue Komponente könnte beispielsweise eine automatische Analyse der Ergebnisse oder die Integration von Sensoren sein [Zel13].

3.1.6. Stabilität und Qualität

Eine Umfrage mit einem bestimmten Probanden wird üblicherweise genau einmal durchgeführt. Daraus ergibt sich die Hauptanforderung der Datenstabilität. Alle erhobenen Daten müssen im System zu jeder Zeit persistent gespeichert sein und dürfen nicht verloren gehen. Darüber hinaus ist die zuverlässige Verschlüsselung nach ISO Standards und Übertragung der Daten an einen Server wichtig, damit diese nicht in die Hände Dritter geraten.

Des weitern muss die Anwendung die Funktion des dynamischen Seitenaufbaus der Fragebögen gemäß der Beschreibungssprache stabil und zuverlässig erfüllen.

3.2. Funktionale Anforderungen

Die funktionalen Anforderungen beschreiben die konkreten Aufgaben des Systems. In dieser Arbeit gehören dazu beispielsweise der Datenschutz oder die Prozessun-

terstützung, die dafür verantwortlich ist, die Ablaufsteuerung eines Fragebogens zu übernehmen.

3.2.1. Mehrsprachigkeit

Der Aspekt der Mehrsprachigkeit ist in dieser Arbeit aus zwei unterschiedichen Seiten zu betrachten, da die Sprache der Anwendung nicht gleich der Sprache des Fragebogens sein muss.

Diese Anforderung wird aus dem folgendem Szenario abgeleitet:
Es wird in Burundi (Afrika) eine Umfrage von erwachsenen Frauen durchgeführt. Diese müssen den Fragebogen selbstständig lesen und beantworten, da Metadaten wie Ausführungsverhalten und Ausführungszeit zusätzlich zu den tatsächlichen Antworten erhoben werden sollen. Es wird davon ausgegangen, dass die zu interviewende Frau keine Deutsch-Kenntnisse aufweist und die betreuende Person keine Kenntnisse der aktuellen Landessprache besitzt. Somit muss die Sprache der Anwendung und die des Fragebogens getrennt voneinander betrachtet werden. In diesem Fall wird zur Vorbereitung und Auswahl des richtigen Fragebogens die Sprache Deutsch verwendet, jedoch im Fragebogen selber die Landessprache der Probandin.

Hierdurch ergibt sich direkt die Anforderung der Pflege neuer Sprachen für die Anwendung und den Fragebogen. Für die Anwendung ist das bereits unter Kapitel 3.1.5 beschrieben. In Bezug auf den Fragebogen muss dies von der Konfiguratorumgebung gepflegt werden. Diese sorgt für die Pflege des kompletten Fragebogen, inklusive der Sprachen und dem Aufbau [Sch14].

3.2.2. Datenschutz und Sicherheit

Im gesamten System ist ein umfassendes Benutzerkonzept angedacht, welches nur ausgewählten Benutzern erlaubt, bestimmte Aktionen im System auszuführen. Die Rechte werden hierbei direkt beim Erstellen der Fragebögen vergeben. Somit ist keine explizite Benutzerverwaltung auf den mobilen Endgeräten notwendig. Lediglich ein administrativer Benutzer wird fest in die Anwendung integriert.

Die Erarbeitung eines Benutzerkonzepts, sowie die Integration mit der mobilen Prozess Engine ist nicht Gegenstand dieser Arbeit.

Aus dem Grund, dass dem System der Inhalt der Fragen nicht bekannt ist, kann man an dieser Stelle nicht ausschließen, dass bei der Umfrage personenbezogene Daten erhoben werden, welche laut Bundesdatenschutzgesetz (§11 BDSG) [MG12] gesondert betrachtet werden müssen. Somit sind in der mobilen Anwendung alle Daten als personenbezoge Daten zu betrachten.

Daten die über diese Anwendung erhoben werden, werden in einer Datenbank auf dem mobilen Endgerät verwaltet und sind ausschließlich über einen ausgewiesenen Administrator Benutzer zugänglich. Nach der Befragung des Probanden soll die Möglichkeit bestehen, mittels einer verschlüsselten Verbindung die erhobenen Daten zum Server zu senden. Durch Einsatz dieser Mechanismen ist die Sicherheit und Integrität der Daten zu gewährleisten.

3.2.3. Dynamische Generierung des Fragebogens

Der Aspekt der dynamischen Generierung des Fragebogens ist hier im Bereich Anforderung gesondert zu betrachten, muss jedoch aufgenommen werden, da es sich um ein Kernbereich dieser Arbeit handelt. Die exakten technischen Anforderungen sind detailiert in den Kapiteln 4 und Kapitel 6 zu finden. An dieser Stelle lässt sich lediglich auf einfache Weise ausdrücken, was die dynamische Darstellung zu leisten hat.

Die Hauptaufgabe der dynamischen Generierung ist das Erstellen einer Benutzeroberfläche mit welcher Interviews durchgeführt werden können. Dazu werden Daten benötigt, welche auf dem Konfigurator erstellt und über eine Webschnittstelle auf das mobile Endgerät übertragen werden. Diese Daten beinhalten eine Beschreibung der darzustellenden Seiten inklusive den Fragen. Auf Grundlage dieser Informationen und unter Einhaltung des Styleguides [Sch14] wird eine Oberfläche zur Interaktion für den Benutzer erstellt. Dabei ist zu beachten, dass auf dem Konfigurator nur Typen von Fragen verwendet werden dürfen, welche auf dem mobilen Endgerät implementiert sind.

3.2.4. Persistieren der erhobenen Daten

Eine weitere wichtige Anforderung ist die Persistierung der erhobenen Daten, also die dauerhafte Speicherung. Die Anwendung, welche auf den mobilen Endgeräten ausgeliefert wird, muss nach Fertigstellung eines Fragebogens in der Lage sein, alle erhobenen Antworten (inklusive der Metadaten) an den Server zu senden. Dieses ist natürlich auch machbar, wenn die Daten im Speicher gehalten werden, jedoch wären diese verloren, falls ein Fehler auftritt oder der Fragebogen abgebrochen wird.

Jede Änderung der Antwort eines Benutzers soll umgehend in einer Datenbank gespeichert werden, damit die Ausfallsicherheit gewährleistet werden kann. Zudem können so bei späteren Analysen nachträgliche Änderungen der Antworten der befragten Person erkannt werden.

3.2.5. Export von Daten

Nachdem ein Interview durchgeführt wurde, müssen die angefallenen Daten zurück auf den Server gesendet werden. Die Daten sollen dort analysiert oder gegebenenfalls weiter als Datei exportiert werden. Dazu implementiert das mobile Endgerät und der Server eine Web Schnittstelle über welche Daten in beiden Richtungen ausgetauscht werden können. Ein direkter Export auf dem mobilen Endgerät ist aus rechtlichen Gründen nicht vorgesehen. Wie in der Abbildung 3.1 zu sehen ist, werden vom Server zum mobilen Endgerät Fragebögen transportiert und in der Gegenrichtung die für diesen Fragebögen erhobenen Antworten.

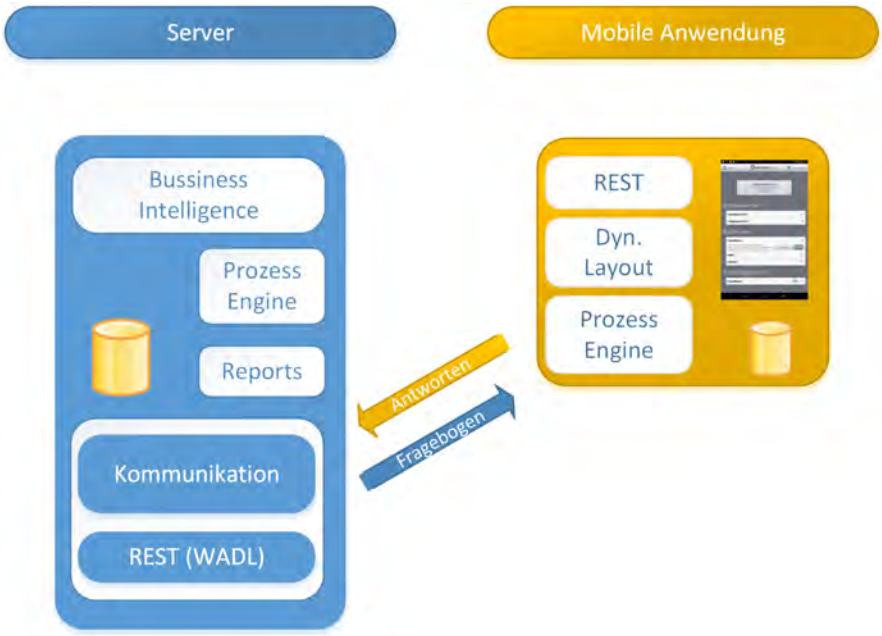


Abbildung 3.1.: Schematisches Kommunikationsmodell zwischen Server und mobilem Endgerät

3.2.6. Parallele Nutzbarkeit

Weiterhin wird gefordert, dass parallel an mehreren mobilen Anwendungen Umfragen zur gleichen Zeit durchgeführt werden können. Die Architektur des Gesamtsystems ist von Grund auf so ausgelegt, dass dies ohne Mehraufwand realisierbar ist. Die Verwaltung der Fragebögen und der Antworten findet zentral auf einem Server statt. Die mobilen Endgeräte dagegen holen sich eine Kopie des auszuführenden Fragebogens und speichern diesen in der lokalen Datenbank. Da auf jedem Gerät eine eigene Prozess Engine vorhanden ist, können die Fragebögen dort zu jeder Zeit völlig autonom ausgeführt werden. Das heißt, es wird keine Internetverbindung und kein zentraler Server mit einer Prozess Engine benötigt um Umfragen durchzuführen, was in Kapitel 2.2 durchaus der Fall war.

Letztendlich werden die gesammelten Antworten von der mobilen Anwendung auf den Server gesendet, welcher jeden ausgefüllten Fragebogen mit einer eigenen ID versieht und in die Datenbank speichert.

3.2.7. Prozessunterstützung

Eine der wichtigsten Anforderungen in dieser Arbeit ist die Existenz einer eigenen Prozess Engine, damit jeder Fragebogen autonom ausführbar ist und die Ausführungslogik als Prozess im Konfigurator modelliert werden kann.

Bei der Realisierung wird eine leichtgewichtige, eigens für diesen Anwendungsfall entwickelte Prozess Engine verwendet, welche im Rahmen einer Projektarbeit entstanden ist. Diese ermöglicht es, komplexe Entscheidungsmuster für Umfragen zu nutzen, ohne die Logik selber Implementieren zu müssen. Das heißt, jeder modellierte Fragebogen kann als Schema mit Ausführungslogik und Inhalt betrachtet werden.

Wie in Abbildung 3.2 zu sehen ist, enthält ein Fragebogen verschiedene Knoten, welche jeweils einer Seite auf dem mobilen Endgerät entsprechen. Jeder Knoten enthält eine Ausführungslogik für die Prozess Engine und eine Seitendefinition für die Anwendung. Des Weiteren ist die Kommunikation ein wichtiger Punkt der Anforderungen. Es muss sichergestellt werden, dass die benötigten Schnittstellen zur Prozess Engine geschaffen werden um Fragebögen zu Verwalten und Interviews durchzuführen.

3.2.8. Offline ausführbar

Wie in der Anforderung im Kapitel 3.2.7 bereits erklärt, befindet sich auf jedem mobilen Endgerät eine eigene Prozess Engine. Zusätzlich sind alle für ein Interview benötigten Daten in der Datenbank gespeichert, falls diese vom zentralen Server angefordert wurden. Mit diesen beiden Voraussetzungen ist es möglich, eine Umfrage zu jeder Zeit und an jedem beliebigen Ort durchführen, ohne eine direkte Verbindung mit dem Internet oder dem eigenen Server zu haben. Diese Anforderung ist von großer Bedeutung, da es in vielen Regionen keine, oder nur eine schlechte Internetverbindung gibt. Eine schlechte Verbindung hätte beispielsweise zur Folge, dass nach dem Beantworten einer Fragebogenseite die Befragung abgebrochen werden muss, weil zum Laden der nächsten Seite ein zentraler Server kontaktiert werden muss. Desweiteren können durch diese Anforderung mobile Endgeräte verwendet werden, welche nur WiFi unterstützen und

somit einen signifikant geringeren Preis haben.

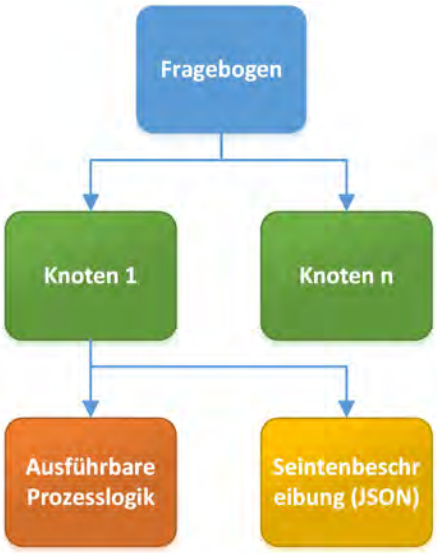


Abbildung 3.2.: Schematischer Aufbau eines Fragebogens

3.2.9. Anforderungen an User Interface Elemente

In diesem Teil wird hauptsächlich auf die UI Elemente eingegangen, welche zur dynamischen Seitenerstellung benötigt werden. Tiefergehende Erklärungen zur Implementierung werden hier nicht behandelt, vielmehr spielt die Art und Weise der Benutzung, sowie psychologische und ergonomische Aspekte eine Rolle. Erklärungen zur Realisierung werden in Kapitel 4 diskutiert.

Wie bereits erwähnt, gibt es einen User Interface Styleguide mit diversen Bedienelementen, welche so im Android Framework nicht zu finden sind. Eine der Hauptaufgaben in diese Arbeit ist es, die spezifizierten Elemente im Android Framework umzusetzen und an den Android Style anzupassen. Hierzu sind für die Umsetzung, mehrere Mockups der geplanten UI Elemente zur Verfügung gestellt worden. Generell sollten jedoch aus Sicht der Usability, bereits besehende UI Elemente der entsprechenden Plattform verwendet

werden, soweit diese die geforderte Funktonalität bieten. Das Implementieren eines neuen Kalenders beispielsweise könnte den Benutzer in seiner Arbeitsweise irritieren, da der neue Dialog an dieser Stelle nicht Erwartungskonform wäre.

Auffallend an diesen Elementen zur Interaktion ist sicherlich, dass jedes Element einen neutralen Wert besitzt. Aus psychologischen Gründen muss das so sein, denn falls bei einem Ja-Nein Schalter bereits ein Wert vorausgewählt ist (also *Ja* oder *Nein*), so nimmt dieser Wert indirekt Einfluss auf die Entscheidung des Probanden. Ihm wird damit suggeriert eine Antwort zu wählen, welche er evtl. aus eigener Entscheidung nicht genommen hätte. Darüber hinaus ist ein neutraler Wert für die Prüfung von Fragen, welche zwingend beantwortet werden müssen, von Vorteil. Jede Antwort, welche nicht dem neutralen Wert entspricht, wurde beantwortet.

Ein weiter Aspekt der generischen UI Elemente ist die Erweiterbarkeit. Alle verwendeten Elemente, sowie das Layouting selber, Framework müssen so modular implementiert sein, dass auf einfache Art und Weise neue Elemente hinzugefügt werden können. Für Umfragen in anderen Domänen könnten weitere Elemente zur Eingabe von Antworten notwendig sein. Durch einen modularen und transparenten Aufbau des UI Layouters kann eine Erweiterung in kurzer Zeit umgesetzt werden. Diese neuen Elemente müssen allerdings auch vom Konfigurator unterstützt werden, da der Aufbau der Fragebögen dort gestaltet wird. Ebenso ist es denkbar bestehende Elemente zu ersetzen, falls sich herausgestellt hat, dass die Akzeptanz des bestehenden Elements unerwartet niedrig ist. Eine Einfachauswahl kann alternativ beispielsweise als Dropdown Box, oder als Liste dargestellt werden.

Im Folgenden werden alle UI Elemente beschrieben, welche für die dynamische Oberflächengenerierung verwendet werden.

Headline

Das Element *Headline* wird verwendet um dem Benutzer eine Überschrift darzustellen. In den meisten Fällen wird es direkt bei der Einleitung des Fragebogens auf der ersten Seite verwendet, kann aber auch an jeder beliebigen Stelle eingesetzt werden. Denkbar

wäre eine *Headline* evtl. noch als visuelle Trennung für einen neuen Abschnitt.
Als weitere Anforderng für dieses Element ist die einheitliche Darstellung von Schrift, Schriftgröße, Schriftart und Style. Diese Attribute sollen im System fest hinterlegt und nicht änderbar sein.

Abbildung 3.3.: Element Überschrift

Text

Das Element *Text* kann zwei visuelle Repräsentationen annehmen. Zum einen als einfacher Fließtext (Abbildung 3.4) und zum anderen als Fließtext mit einem hinterlegten Paragraphen (Abbildung 3.5). Im Gegensatz zur vorherigen Headline soll es hier explizit möglich sein, die Formatierung des Textes zu beeinflussen. Beispiele für mögliche Formatierungen wären fett, kursiv, unterstrichen und das hinzufügen von Absätzen.
Das Text Element dient an sich nur zu Darstellung von Inhalt und kann nicht direkt zur Beantwortung von Fragen verwendet werden, da es keine Möglichkeit zur Eingabe oder Änderung des Textes bietet. Aus diesem Grund wird hier kein neutraler Zustand bereitgestellt.

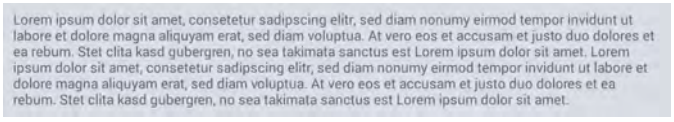


Abbildung 3.4.: Element Text

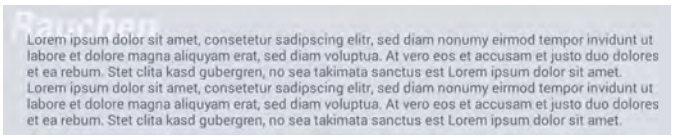
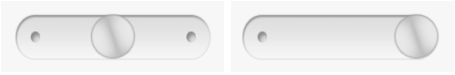


Abbildung 3.5.: Element Text mit Paragraph

Zweistellungschalter (Yes-No Slider)

Der *Zweistellungsschalter* wird dazu verwendet, dem Benutzer genau zweit vordefinierte Werte zur Auswahl anzubieten. Initial ist der Schalter in der neutralen Position verankert, welcher sich in der Mitte des Elements befindet (siehe Abbildung 3.6 a)). Nachdem das erste Mal vom Benutzer ein Wert ausgewählt wurde, also der Schalter nach links oder rechts bewegt worden ist, ist es nicht mehr möglich den neutralen Zustand auszuwählen (Abbildung 3.6 b)). Alle möglichen Endpositionen werden dem Benutzer visuell durch zwei kleine dunkle Punkte links und rechts des Instruments angezeigt.



(a) Neutral (b) Ausgewählt

Abbildung 3.6.: Zweistellungsschalter

Dreistellungsschalter mit neutraler Position (Yes-Neutral-No Slider)

Wenn man sich bezüglich einer Frage enthalten möchte, ist ein weiterer dritter Zustand notwendig. Dieser Zustand wird in der hier dargelegten Arbeit als *neutraler Zustand* bezeichnet und ist bei diesem Interaktionselement vor ausgewählt. Der *Dreistellungsschalter* ist gleich aufgebaut, wie der *Zweistellungsschalter*, besitzt jedoch im Zentrum die zusätzliche dritte Position (siehe Abbildung 3.7 a)).
Im Unterschied zum vorherigen Instrument ist es hier möglich, die neutrale Position erneut anzuwählen, nachdem bereits einen anderen Wert selektiert wurde. Die zusätzliche Position wird dem Benutzer hierbei durch einen weiteren dunklen Punkt im Zentrum dargestellt (siehe Abbildung 3.7 b)).

(a) Neutral (b) Ausgewählt

Abbildung 3.7.: Dreistellungsschalter

Mehrstellungsschalter mit neutraler Position (Slider)

Bei manchen Fragen reichen drei vordefinierte Zustände nicht aus, beispielsweise bei der Frage nach dem Geburtsmonat. Hierbei wäre es denkbar, den Dreistellungsschalter um weitere Zustände zu erweitern, jedoch ergeben sich bei einer größeren Anzahl von Werten Platzprobleme. Deshalb wird hierfür das neues Interaktionselement *Mehrstellungsschalter* eingeführt. Anhand dieses Elementes kann der Benutzer mehrere vordefinierte Werte anhand einer horizontal angedeuteten Skala auswählen. Auch hier befindet sich der Schieberegler initial im neutralen Zustand auf der linken Seite des Instruments (Abbildung 3.8). Dieser Zustand ist für den Benutzer zu jeder Zeit der Umfrage erneut selektierbar, auch nachdem bereits ein konkreter Wert ausgewählt wurde. Eine besondere Funktion in diesem Instrument hat der linke dunkelgrau abgesetzte Bereich vor dem ersten konkreten Wert. Dieser soll eine Art Rampe symbolisieren, über welche der Schieberegler bewegt werden muss um den ersten konkreten Wert zu erreichen. Dieses doch etwas ungwöhnliche Verhalten des Elements wird dem Benutzer so verdeutlicht, dass der Regler im Bereich der Rampe „schwerer“ zu führen ist, als auf der restlichen Skala. Bei einer Schiebebewegung von links nach rechts bewegt sich der Regler im Rampenbereich langsamer als der Finger auf dem Touch Screen. In umgekehrter Richtung bewegt sich der Regler automatisch auf die linke Seite (neutrale Position), falls der Rampenbereich erreicht wird.

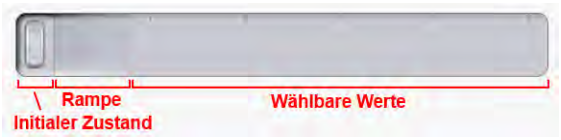


Abbildung 3.8.: Element Mehrstellungsschalter in neutralem Zustand

Text Eingabefeld (Text Input)

Das Element *Text-Eingabe* wird zur Eingabe von Freitexten mittels einer Tastatur verwendet. Der neutrale Wert enspricht hier einem leeren Textfeld. Um das Löschen eines bereits eingegebenen Textes einfacher zu gestalten, wird an der rechten Seite ein Taster angebracht, welcher bei Betätigung den kompletten Inhalt des Textelements löscht (Abbildung 3.9).
Als Editor soll hier die Plattform spezifische Tastatur gewählt werden. Exemplarisch wird in dieser Arbeit die Tastatur einer Android Plattform verwendet, kann aber beliebig ausgetauscht oder abgeändert werden, um eigens erstellte Editoren zu verwenden.

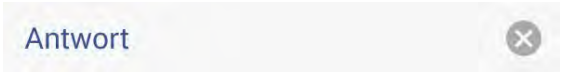


Abbildung 3.9.: Element zur Eingabe von Text

Nummern Eingabefeld (Number Input)

Das Element *Nummern-Eingabe* ist gleich aufgebaut, wie das *Text-Eingabe* Element und bietet dem Benutzer die gleichen Funktionen. Jedoch wird hier der Plattform spezifische Eingabedialog für Ziffern verwendet.

Datum Eingabefeld (Date Input)

Mit dem Element *Datum-Eingabe* soll es möglich sein, mittels eines Editors ein frei wählbares Datum einzugeben. Um ein bereits eingegebenes Datum zu Löschen, damit beispielsweise der neutrale Zustand erreicht wird, befindet sich auch hier ein Taster auf der rechten Seite, welcher bei Betätigung den kompletten Inhalt des Elements löscht.
Als Editor soll hier ein *Pop-up* verwendet werden, welches dem Benutzer einen Kalender visualisiert. Auch hier kann bereits auf Plattform spezifische Elemente zurückgegriffen

werden. In Abbildung 3.10 ist exemplarisch ein Editor mit einem Kalender als Standard Element des Android Frameworks zu sehen.

Abbildung 3.10.: Editor zur Eingabe von Datum

Bereichsauswahl (Range Selection)

Mit dem *Bereichsauswahl* Element soll es auf einfache Art und Weise ermöglicht werden, einen oder mehrere Teilbereiche aus einem vorgegebenen Gesamtbereich auszuwählen. Die ausgewählten Bereiche werden zur besseren Übersicht für den Benutzer als vertikale Balken auf einer horizontalen Skala visualisiert (Abbildung 3.11). Der neutrale Wert wär hier gegeben, falls kein Wert im Wertebereich selektiert ist.

Als Editor dient wiederum ein *Pop-up*, welches Taster für jeden Wert im vorgegebenen Bereich enthält (Abbildung 3.12). Ein selektierter Taster soll dabei im gedrückten Zustand so lange verharren, bis er wieder deselektiert wird. Auf diese Weise kann der Benutzer sehr einfach Bereiche innerhalb des gegebenen Intervalls bearbeiten.

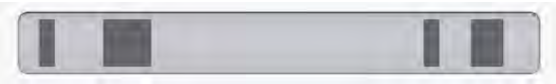


Abbildung 3.11.: Element Bereichsauswahl

Abbildung 3.12.: Editor zur Eingabe von Bereichen

Einfachauswahl (Single Choice)

Mit dem Element *Einfachauswahl* soll genau ein Wert aus einer Liste von Werten ausgewählt werden. Der neutrale Wert wird dabei auf der Oberfläche mit drei Strichen codiert.
Dieses Element könnte durch verschiedenste visuelle Repräsentationen dargestellt werden, wurde jedoch aus zeitlichen Gründen im Dropdown-Box Style implementiert (Abbildung 3.13). Alternative Repräsentationen wären beispielsweise eine komplett sichtbare Liste mit den vorgegebenen Werten, aus dieser genau ein Wert ausgewählt werden kann, oder eine Repräsentation als Spinner.

Abbildung 3.13.: Element Einfachauswahl mit geöffnetem Editor

Mehrfachauswahl / Fragegruppe

Wie in der Abbildung 3.14 zu erkennen ist, werden für das Element der *Mehrfachauswahl* bereits beschriebene Elemente zusammengefasst und in zwei unterschiedliche Bereiche unterteilt. Der obere Bereich besteht üblicherweise aus Zwei- oder Dreistellungsschaltern (siehe Kapitel 3.2.9 und 3.2.9) der untere Bereich aus einem Textfeld für eine alternative textbasierte Antwort.
Jede Auswahlmöglichkeit wird hierbei als einzelne Frage behandelt. Die gesamte Gruppe ist allerdings erst beantwortet, wenn für jede zwingend notwendige Frage eine Antwort vorliegt. Die notwendigen Fragen sind mit einem Asterisk (*) gekennzeichnet (Abbildung 3.14).

Abbildung 3.14.: Element Mehrfachauswahl / Fragegruppe

Frageblock

Mit dem Element *Frageblock* können einzelne Fragen, welche einen thematischen Bezug zueinander haben, zu einem logischen Block zusammengefasst werden. Dabei können alle bisher beschriebenen Einzelelemente verwendet und beliebig kombiniert werden. In der Abbildung 3.15 sind exemplarisch eine Datumseingabe, ein Zweistellungsschalter und ein Text Element dargestellt. Auch hier ist der Fragenblock erst beantwortet, wenn für jede zwingend notwendige Frage eine Antwort vorliegt. Die notwendigen Fragen sind

mit einem hochgestellten Asterisk (*) gekennzeichnet. Im Unterschied zu *Fragengruppe* kann hier kein zusätzlicher alternativer Text eingegeben werden.

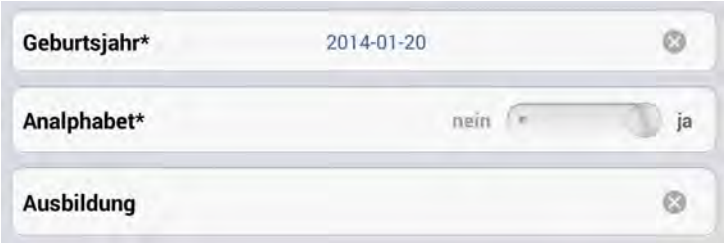


Abbildung 3.15.: Element Fragegblock

3.2.10. Kontroll Elemente zum Blättern durch den Fragebogen

Zentraler Punkt des Fragebogensystems ist die Validierung der Antworten. Dazu kann der Button zur Weiterschaltung auf die nächste Seite des Fragebogens zwei Zustände annehmen. Zu Beginn jeder Seite ist er nicht bedienbar und mit grauem Hintergrund dargestellt (Abbildung 3.16 a)). In diesem Zustand bleibt er so lange, bis für alle Fragen, welche zwingend beantwortet sein müssen, eine Antwort vorliegt. Erst dann wird der Button mit blauem Hintergrund dargestellt, und kann bedient werden. (Abbildung 3.16 b)).Somit ist ein Blättern auf die nächste Seite erst möglich, wenn alle Antworten, die zwingend vorliegen müssen, auch tatsächlich vorhanden sind.

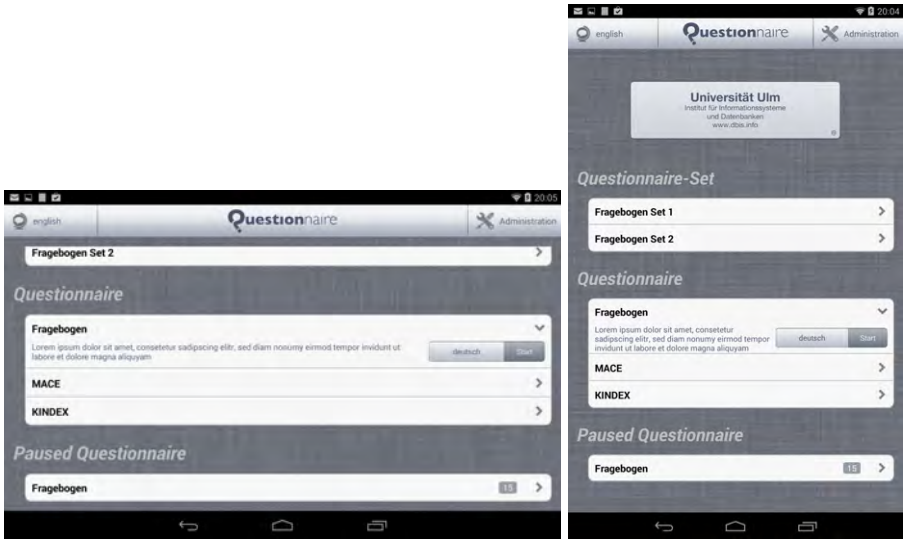
Die dazu gehörende Validierung der Antworten muss bei jeder Wertänderung der Antworten stattfinden, weil eine bereits erhaltene Antwort unter Umständen mittels des neutralen Wertes wieder zurückgenommen werden kann.

(a) Deaktivierter Zustand (b) Aktivierter Zustand

Abbildung 3.16.: Weiter Button

3.2.11. Änderung der Bildschirm-Orientierung

Das Layout der Anwendung soll so gewählt und implementiert werden, dass die Änderung der Ausrichtung von *Hochformat* auf *Querformat* und zurück untertstützt wird. In den folgenden Abbildungen 3.17 a) und b) wird dieses Verhalten exemplarisch verdeutlicht.



(a) Querformat (b) Hochformat

Abbildung 3.17.: Startbildschirm in verschiedenen Orientierungen

4

Konzept

In diesem Kapitel wird das Konzept der Anwendung schematisch betrachtet. Dabei wird keine Bezug auf konkret Implementierungsaspekte in Verbindung mit dem Android Framework genommen. Hinweise, Erläuterungen und Implementierungsaspekte sind ausschließlich in Kapitel 6 zu finden. Im Kapitel 4.1 werden allgemeine Themen und grundsätzliche Überlegungen zur Anwendung und dem Konzept behandelt, welche als Voraussetzungen für das eigentliche Systemmodel zu sehen sind. Kapitel 4.2 werden die Hauptkomponenten der Anwendung vorgestellt und beschrieben. Der Aufbau eines Fragebogens wird in Kapitel 4.3 behandelt. In den Kapiteln 4.4 bis 4.6 wird auf die wichtigsten Komponenten der Systemarchitektur näher eingegangen, wie beispielsweise die Prozess Engine oder der User Interface Generator. Einen Überblick der unterstützten Editoren der Anwendung wird in Kapitel 4.7 gegeben. Schlussendlich wird in Kapitel 4.8 auf den Aspekt der Mehrsprachigkeit eingegangen.

4.1. Grundsätzliche Überlegungen und Voraussetzungen

In diesem Kapitel werden die Voraussetzungen für das eigentliche Konzept diskutiert. Dazu gehören beispielsweise die Wahl des Betriebssystems, oder die Realisierungsform der Anwendung.

4.1.1. Betriebssystem

Zum Zeitpunkt der Realisierung dieser Arbeit gelten die Betriebssysteme Android, iOS und Windows als die am weitest verbreiteten Plattformen für mobile Endgeräte. Von der expliziten Auswahl eines Betriebssystems zur Realisierung der mobilen Anwendung kann jedoch abgesehen werden, da durch die Formulierung des Titels dieser Arbeit die Android Plattform bereits gesetzt ist. Realisiert wurde die Anwendung unter Verwendung der Version 4.4, darum wird diese oder eine höhere Version initial auf dem mobilen Endgerät gefordert.

An dieser Stelle wird eine kurze Übersicht über die Vor- und Nachteile der gewählten mobilen Plattform diskutiert.

Vorteile: Die weite Verbreitung der Android Plattform unter den verschiedenen Herstellern bietet dem Markt vielfältige Hardware in unterschiedlichen Preisklassen. Dadurch ist eine Vielzahl von Geräten mit sehr gutem Preis-Leistungs-Verhältnis zu finden.

Ein weiterer Vorteil der meisten Geräte ist die Existenz eines USB Ports, welcher hauptsächlich zum Laden vorgesehen ist. Jedoch ist es auch möglich eine Tastatur, oder verschiedene externe Geräte anzuschließen.

Das System selber ist sehr performant und wird stetig weiterentwickelt. Für Entwicklungseinsteiger bietet die Bereitstellung einer SDK mit umfangreichen Bibliotheken große Unterstützung.

Den Android Ein- oder Umsteigern wird es in Bezug auf die Bedienung der Software und des Geräts sehr einfach gemacht. Alle Oberflächen inklusive ihrer dargestellten Funktionen sind weitgehend selbsterklärend und benötigen keine Schulung im Vorfeld.

Durch einfaches *Spielen* mit dem bereitgestellten User Interface findet man sich sehr schnell in die Bedienphilosophie ein.

Nachteile: Durch die Vielfalt an Geräten und Herstellern müssen bei der Programmierung verschiedenste Auflösungen und Formate des Bildschirms unterstützt werden, damit eine fehlerfreie und einheitliche Darstellung gewährleistet ist.

Ebenso der Herstellervielfalt geschuldet, muss auf ein Update des Betriebssystems mitunter lange Zeit gewartet werden. Ein von Google bereitgestelltes Update muss von jedem Hersteller auf die eigene Geräteinfrastruktur anpasst werden, bevor es verwendet werden kann. Dies ist mit erheblichem Aufwand verbunden und wird deshalb nicht regelmäßig von den entsprechenden Geräteherstellern durchgeführt.

Bedingt dadurch, dass Android ein Open Source Betriebssystem ist und die Anwendungen in den verschiedensten Märkten nicht auf ihre tatsächliche Funktionalität überprüft werden, besteht die Gefahr Schadsoftware zu erhalten.

In einer aktuelle Studie über die Marktanteile der mobilen Betriebssysteme für Smartphones erreichte Android im ersten Quartal 2013 einen Wert von 75% (Abbildung 4.1). [Gar14] Durch dieses Ergebnis wird sichergestellt, dass mit der Wahl von Android auf ein zukunftssicheres Betriebssystem gesetzt wurde.

4.1.2. Mobile Endgeräte

Über die Verwendung von bestimmten Endgerät-Klassen wurden keine direkten Restriktionen gelegt. Vielmehr spielt die Größe des Bildschirms und die Performance des Prozessors eine Rolle.
Geräte mit zu kleinen Displays, wie die meisten gängigen Smartphones, werden für die Anwendung nicht in Betracht gezogen. Deshalb wurde festgelegt, dass die Größe des Bildschirmes größer oder gleich 7 Zoll sein sollte. In dieser Anwendung wird die Android spezifische Auflösung `xhdpi` unterstützt. Des Weitern spielt die Performance bei der dynamischen Generierung der Seiten eine erhebliche Rolle, wodurch nur Tablet PC's mit

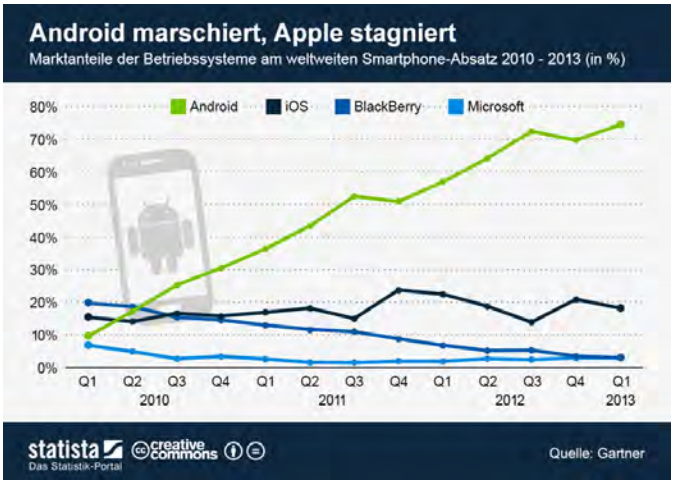


Abbildung 4.1.: Marktstudie - Mobile Betriebssysteme [Gar14]

einem geeignetem Prozessor verwendet werden sollten. Bei der Entwicklung hat sich das Google Nexus 7 (2 Generation) als sehr brauchbar herausgestellt.

4.1.3. Native Anwendung oder Webanwendung

Der grundlegende Unterschied zwischen einer nativen Anwendung und einer Webanwendung, liegt darin, dass native Anwendungen speziell auf ein Betriebssystem (iOS, Android, Windows Mobile) oder sogar auf bestimmte Geräte zugeschnitten sind. Im Gegensatz dazu funktioniert eine Webanwendung potenziell auf jedem Endgerät, welches über den standardkonformen Browser verfügt.

Webanwendung: Eine Webanwendung wird üblicher Weise über den Browser aufgerufen und ist im Prinzip eine Anwendung, die mit Web-Technologie erstellt wurde. Anhand des Aussehens oder der Bedienung lässt sie sich meist nicht von einer nativen Anwendung unterscheiden.

Webanwendungen haben gegenüber nativen Anwendungen den Vorteil, dass nur eine Anwendung entwickelt werden muss, die alle Plattformen bedient. Änderungen an der Webanwendung sind vergleichsweise einfach zu bewerkstelligen. Das Ausrollen eines Updates ist simultan für alle Endgeräte verfügbar. Dafür ist der

Zugriff auf Gerätefunktionen, wie beispielsweise Datenbanken, nur eingeschränkt möglich.

Native Anwendung: Native Anwendungen sind Programme, welche direkten Zugriff auf Hardware-Funktionen des Endgeräts haben, wodurch die volle Mächtigkeit der verwendeten Plattform genutzt werden kann. Beispielsweise können Daten des GPS-Moduls oder des Lagesensors verwendet werden. Dies ermöglicht eine Reihe von Anwendungen, welche mit dem Webansatz nicht realisierbar sind. Die Entwicklung von nativen Anwendungen findet für jedes Betriebssystem dediziert statt. Wer iOS, Android & Windows bedienen möchte, muss also drei verschiedene Anwendungen bereitstellen. Dementsprechend aufwendig ist auch die Durchführung von Updates.

Zusammenfassend lässt sich sagen, dass die Entwicklung nativer Anwendungen zu bevorzugen ist, wenn die Anwendung über eine komplexe Nutzeroberfläche verfügen soll, welche die Nutzung spezieller Hardwarefunktionalitäten erfordert oder die Performance der Anwendung einen kritischen Faktor darstellt. Immer wenn diese Punkte nicht zutreffen, ist die Erstellung einer Webanwendung durch ihre schnellere Entwicklungszeit, den geringeren Kosten und der Möglichkeit, höherer Update-Frequenzen eine mehr als ernstzunehmende Alternative.

Unter Berücksichtigung dieser Punkte stellt sich sehr schnell heraus, dass die Anforderung über die Existenz einer eigenen Prozess Engine zur Ausführung von Fragebögen und der damit verbundenen Möglichkeit, Fragebögen auch ohne Internetverbindung auszuführen nur dann nachkommen können, wenn eine native Anwendung konzipiert wird. Nachfolgend werden ausgewählte Aspekte betrachtet, welche zur Entscheidung beigetragen haben.

Eigene Prozess Engine: Der Browser ist in gewisser Weise eine Sandbox, und kann somit nicht auf beliebige Ressourcen und Programme des Betriebssystems zugreifen. Somit müsste bei einer Webanwendung die Prozess Engine als Plugin implementiert, und die Existenz einer Datenbank auf dem mobilen Endgerät vorausgesetzt werden. Dadurch ist der Aspekt der Plattformunabhängigkeit bei We-

banwendungen nicht mehr gegeben, da das Prozess Engine Plugin und eine passende Datenbank auf dem Endgerät vorhanden sein muss. Bei Verwendung der nativen Anwendung als Realisierungsform ist die Prozess Engine bereits in der Anwendung vorhanden und kann direkt angesprochen werden.

Aufwendige Grafik: Ein weiteres Problem bei Webanwendungen ist die Performance bei der Darstellung von komplexen Benutzeroberflächen mit sehr vielen ineinander geschachtelten Elementen. Der aktuelle Stand der Browser-Implementierungen erlaubt es nicht direkt auf die Grafikkarte zuzugreifen. Dadurch wird die komplette Benutzeroberfläche nicht vom dafür ausgelegten Grafikprozessor (GPU) gerendert, sondern fällt auf den zentralen Prozessor (CPU) zurück. Um einen möglichst generischen und erweiterbaren Ansatz zu garantieren, ist dies in der zu realisierenden Anwendung so gelöst (siehe Kapitel 4.6).

Sensoren: In der hier dargelegten Arbeit ist es angedacht, die erhobenen Antworten mit zusätzlichen Daten von verschiedensten Sensoren ([SSP+13]) anzureichern. Bei einer Webanwendung ist es nicht möglich, die volle Mächtigkeit des Android Betriebssystems zu nutzen, da für die Schnittstellen zwischen Browser und Betriebssystem einige Restriktionen gelten. Somit ist die Verwendung von Sensoren nur eingeschränkt möglich.

Die konkrete Wahl der Plattform wurde aus praktikablen Gründen auf Android festgelegt, da diese aktuell auch die größte Verbreitung aufweist [Gar14].

4.1.4. Design

Wie bereits erwähnt wurde, existiert für die Oberfläche ein Styleguide, welcher das Design der Anwendung im Details beschreibt und festlegt. Die Umsetzung dieser Vorgaben wird ein Kernpunkt der Realisierung darstellen, da dabei einige wichtige Aspekte zu beachten sind.

Keine Standard-Elemente: Die Designvorgabe enthält speziell für Umfragen auf mobilen Endgeräten entworfene Interaktionselemente, welche in dieser Form im Android Framework nicht enthalten sind. Deshalb müssen diese Elemente auf Basis der

Android Plattform nachgebildet werden. Ein solches Interaktionselement ist beispielsweise ein Zweistellungsschalter mit einem zusätzlichen neutralen Zustand, welcher initial vorausgewählt ist (siehe Kapitel 4.6).

Urheberrecht: Einige in der Designvorgabe enthaltenen Interaktionselemente sind von der Firma Apple Urheberrechtlich geschützt, da die Vorgabe auf iOS Basis erstellt wurde. Dazu gehört beispielsweise das Design des *Spinning Wheel*, welches zur Auswahl von Datum eingesetzt wird. Diese Interaktionselemente müssen abgeändert, oder alternative Repräsentationen gefunden werden, falls sie nicht im iOS Kontext verwendet werden.

Technische Herausforderungen: Das Umsetzen eines Layouts, welches auf Papier entworfen wurde, stellt meistens eine besondere Herausforderung dar. Vor allem dann, wenn die enthaltenen Elemente auf Basis eines Betriebssystems mit abweichender Philosophie in der Bedienung erstellt wurde. Daraus ergibt sich die Herausforderung, passende Elemente und Container zu finden, um den Designvorgaben gerecht zu werden.

4.1.5. Kommunikation mittels REST

Representational State Transfer (REST) [Fie14] bezeichnet ein Programmierparadigma zu Kommunikation zwischen Anwendungen. Es gibt keine explizite Norm, daher gehen die Vorstellungen, was REST ist, auseinander. Im Grunde bezeichnet REST die Idee, dass eine URI genau einen Seiteninhalt als Ergebnis einer serverseitigen Aktion (etwa das Anzeigen einer Trefferliste nach einer Suche) darstellt, wie es der Internetstandard HTTP für statische Inhalte bereits vorsieht. Dieses Ziel, erfordert für dynamisch erzeugte Seiten mitunter jedoch zusätzlichen Aufwand.

Zu Beginn des Gesamtprojekts wurden für die Kommunikation zwischen den Teilsystemen Konfigurator, Server und mobiles Endgerät drei weit verbreitete REST Frameworks untersucht. Um nicht nur theoretische Aspekte vergleichen zu können, galt es als Aufgabe für jedes dieser Kommunikationsumgebungen ein kleines Testprojekt zu erstellen. Verglichen wurden letztendlich folgende Aspekte:

Dokumentation: Die Dokumentation ist ein wichtiger Aspekt bei der Wahl des Kommunikationsframeworks. Eine vernünftige Dokumentation zu lesen ist wesentlich einfacher als beispielsweise vorhandenen Code zu analysieren. Zudem vermittelt die Dokumentation einen Überblick über den Aufbau des gesamten Systems. Zusätzlich kann die Funktionsweise einzelner Methoden recherchiert werden. Dabei ist verstärkt auf die Güte der Dokumentation zu achten.

Funktionsumfang: Der Funktionsumfang ist ein sehr interessanter Aspekt, denn viele Frameworks bieten nur Basisfunktionalität. Beispielsweise nur Methoden zum Senden und Empfangen von Daten, jedoch keine zusätzlichen Methoden um die Nachricht aufzubauen. Ein wichtiges Kriterium im Gesamtprojekt ist die Verarbeitung und Erzeugung einer WADL.

Die Web Application Description Language (WADL) ist ein XML-basiertes Dateiformat, welches die Schnittstellen von HTTP-basierte Anwendungen in maschinenlesbarer Form beschreibt. WADL unterstützt dabei vor allem die Beschreibung von REST-basierten Webservices und nimmt für diese Klasse von Anwendungen eine vergleichbare Rolle ein wie WSDL für SOAP-basierte Webservices [MJH09].

Architektur: Mit diesem Aspekt Architektur wird auf den internen Aufbau des Frameworks eingegangen. Hierbei wird beispielsweise überprüft, ob es Erweiterbar ist, oder konsistent in einer Programmiersprache realisiert.

Kopplung: Bei diesem Aspekt wird geprüft, ob das Framework alle gestellten Anforderungen in Bezug auf das Gesamtsystem erfüllt. Ist das Framework beispielsweise für alle im Gesamtsystem eingesetzten Programmiersprachen verfügbar.

Performance: Die Performance bezieht sich in diesem Kontext zum einen auf das interne Laufzeitverhalten des Frameworks selbst. Zum anderen auf die Übertragungsdauer beim Empfangen und Senden von Daten.

Wartung / Weiterentwicklung: Dieser Aspekt ist von besonderer Bedeutung in Bezug auf die Realisierung. Denn nur wenn das Framework gewartet wird, können unerwartet auftretende Softwarefehler gemeldet und behoben werden. Zugleich ist es von Vorteil auf aktuelle technologische Fortschritte schnell zu reagieren und das Framework weiter zu entwickeln.

Bei der Auswertung (siehe Tabelle 4.1) kristallisierte sich anfangs das Play Framework als Spitzenreiter heraus, da es in den Aspekten Dokumentaktion und Funktionsumfang sehr weit fortgeschritten ist. Schlussendlich lief die Wahl jedoch auf das Jersey Framework, da es in allen Kategorien gut abgeschnitten hat, und Play nicht konsistent in Java realisiert wurde (beispielsweise wurden Teile in *Scala* umgesetzt). Die Bewertung der Performance wurde letztendlich außen vor gelassen, da die gemessenen Daten des rudimentären Testprojekts nicht wirklich repräsentativ sind. Weiter spielt die Performance bei der Übertragung der Daten in diesem Projekt eher eine untergeordnete Rolle, da die REST Schnittstelle nur zum Transport der Fragebögen vom Server zum mobilen Endgerät und zur Rückmeldung der erhobenen Daten verwendet wird. Das Durchführen von Interviews findet Offline statt, somit wird die Schnittstelle während der Befragung nicht benötigt.
Der ausschlaggebende Punkt bei der Wahl des Jersey Frameworks war der Support einer WADL (Web Application Description Language). Diese Enthält die Definition der kompletten Webschnittstelle. Jersey unterstützt dabei sowohl das Lesen, als auch das Schreiben einer WADL.

Kriterien	Jersey	Restlet	Play
Dokumentation	+	−	++
Funktionsumfang	+	+	++
Architektur	+	+	0
Kopplung	+	+	
Performance	(+)		
Wartung	+	0	0

Tabelle 4.1.: Entscheidungsmatrix REST Frameworks

4.1.6. Persistieren der Daten

Das Thema der Persistierung hat letztendlich mehr Zeit in Anspruch genommen, als zuvor gedacht. Zu Beginn des Projekts waren die Anforderung an diesen Punkt noch nicht transparent. Initial wurde von einer verteilten Speicherung ausgegangen. Das bedeutet im Kontext dieser Arbeit, dass Teile des Fragebogens und der Antworten sowohl

von der Prozess Engine als auch von der Anwendung selber verwaltet werden müssen. Der Nachteil einer verteilten Speicherung ist die teilweise doppelte Datenhaltung sowie die doppelte Implementierung der Datenbankfunktionen. Um dem entgegen zu wirken wurde später entschieden, die Prozess Engine als zentrale Schnittstelle zur Datenbank zu verwenden. Diese bietet Methoden zum Verwalten von Fragebögen, zum Starten einer Befragung und zur Speicherung der erhobenen Daten an. Darüber hinaus wird so eine saubere Struktur im Bereich Code- und Funktionstrennung erreicht, welche die Anforderung der Wartbarkeit signifikant erhöht.

4.2. Systemarchitektur

In diesem Kapitel werden die Hauptkomponenten der zu realisierenden Anwendung vorgestellt und beschrieben. Die Abbildung 4.2 zeigt eine schematische Übersicht der Systemkomponenten.

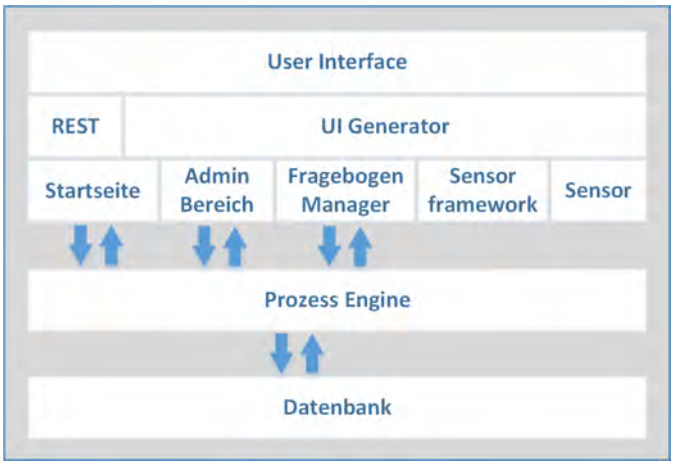


Abbildung 4.2.: Systemarchitektur der Anwendung

REST: Die Anwendung implementiert eine REST Schnittstelle, mit welcher vom Server Fragebögen angefragt, und beantwortete Fragebögen zurück geschickt werden können. Die Definition der Webschnittstelle erfolgt über die WADL (Web Application

Description Language). Gründe für die Wahl des REST Frameworks wurden bereits in Kapitel 4.1.5 diskutiert.

Startseite: Die Startseite ist der Einstiegspunkt der Anwendung. Diese dient zur Navigation auf den administrativen Bereich und enthält eine Funktion zur Wahl der Sprache der Anwendung. Auf der Startseite können Fragebögen aus einer Liste ausgewählt, deren Sprache eingestellt und gestartet werden.

Administrativer Bereich: Der administrative Bereich ist über eine zusätzliche Benutzer-Authentifizierung erreichbar. Dieser bietet Funktionen zum Verwalten von Fragebögen. Des Weiteren können dort die Antworten der Fragebögen zurück an den Server gesendet werden. Der administrative Bereich enthält zusätzlich Systemeinstellungen, wie bespielsweise Benutzername, Passwort und URI des Servers.

Fragebogenmanager: Der Fragebogenmanager ist für die Bereitstellung der Fragebogenseite verantwortlich, welche vom User Interface Generator erstellt wird. Des Weiteren verwaltet er die erhobenen Antworten und stellt Funktionen zur Navigation zwischen den Seiten des Fragebogens zur Verfügung. Die Steuerung der Reihenfolge übernimmt dabei die Prozess Engine. Ferner verwaltet der Fragebogenmanager zusätzliche vom Sensorframework bereitgestellte Metadaten.

User Interface Generator: Die Aufgabe des User Interface Generators ist es eine dynamisch generierte Benutzeroberfläche zur Beantwortung der Fragebögen bereit zu stellen. Dazu benötigt er eine Seitenbeschreibung, welche in der Konfigurationsumgebung erstellt wird. Eine Seitenbeschreibung enthält im wesentliche die darzustellenden Seiten, die dazugehörigen Fragen und die verwendeten Textressourcen in einer strukturierten Form.

User Interface: Das User Interface bietet dem Benutzer eine Bedienoberfläche zur Interaktion. Mit dieser kann kann er Fragebögen auswählen, starten und beenden. Die meisten Interaktionselemente zur Beantwortung von Fragen wurden eigens für diesen Anwendungsfall realisiert.

Sensorframework: Das Sensorframework dient zu Kommunikation mit verschiedenen Sensoren, die am mobilen Endgerät angeschlossen oder direkt verbaut sind.

Damit können die vom Benutzer erhobenen Antworten um zusätzliche Daten, wie beispielsweise Vitalparameter angereichert werden [Zel13].

Sensor: Sensoren können direkt auf dem mobilen Endgerät integriert sein (beispielsweise Kamera oder Gyroskop) oder über eine Schnittstelle angebunden. Externe Sensoren (beispielsweise Pulsmesser) kommunizieren dabei meist über eine Kabelverbindung oder Bluetooth. Das Sensorframeworks [Zel13] abstrahiert dabei die verschiedenen Kommunikationstypen und bietet der Anwendung eine einheitliche Schnittstelle zur Verwendung der verfügbaren Sensoren.

Prozess Engine: Die Prozess Engine ist eine zentrale Komponente in dieser Anwendung und bietet Schnittstellen zum Verwalten von Fragebögen und deren Antworten. Dabei hat sie als einzige Komponente eine direkte Schnittstelle zur Datenbank, mit welcher alle zum Prozess gehörenden Daten persistiert werden. Die Reihenfolge der darzustellenden Fragebogenseiten wird von der Prozess Engine bestimmt, somit übernimmt sie die Steuerung eines Interviews.

Datenbank: Die relationale Datenbank befindet sich direkt auf dem mobilen Endgerät und ist für das Speichern der Fragebögen und deren Antworten verantwortlich.

4.3. Struktureller Aufbau eines Fragebogens

Im Kontext der zu realisierenden Anwendung kann ein Fragebogen als Schema zur Erstellung und Durchführung eines Fragebogens betrachtet werden. Ein Fragebogen enthält dazu mehrere Knoten, welche die kleinsten Einheiten darstellen. Jeder Knoten wird bei der Modellierung in der Konfiguratorumgebung mit zusätzlichen Informationen angereichert und repräsentiert dabei eine Seite des Fragebogens. Ein Knoten besteht dabei im Wesentlichen aus Seitenbeschreibung und Ausführungslogik.

4.3.1. Seitenbeschreibung

Damit auf dem mobilen Endgerät eine dynamisch erzeugte Benutzeroberfläche dargestellt werden kann, muss der Anwendung eine Beschreibung der darzustellenden

Seite vorliegen. Diese Informationen sind in der Seitenbeschreibung enthalten, welche in der Konfiguratorumgebung modelliert wird. Die Abbildung 4.3 zeigt die wichtigsten Elemente dieser Struktur. Das oberste Element ist dabei eine Seite (`page`), welche aus einer Liste von Elementen (`pageElements`) besteht. Ein `pageElement` repräsentiert dabei jeweils einen Abschnitt auf der Benutzeroberfläche. Unterschieden werden diese Abschnitte anhand des `type` Attributs (beispielsweise `HEADLINE` oder `QUESTION`). Die visuelle Repräsentation dieser Abschnitte wird im Kapitel 4.6 veranschaulicht.

Ein `questionElement` repräsentiert jeweils eine Frage auf der Benutzeroberfläche. Unterschieden werden die Fragen anhand des Attributes `questionType` (beispielsweise `BOOLEAN` oder `SLIDER`). Alle mit Asterisk (*) gekennzeichneten Felder auf der Abbildung 4.3 sind Container für Übersetzungen (`languageTextContainer`) und enthalten Textressourcen für die unterstützten Sprachen.

Interne Repräsentation einer Seite

Ein weiterer wichtiger Punkt des Konzeptes ist die anwendungsinterne Repräsentation der in JSON codierten Seitenbeschreibung. Dazu wird anhand der Seitenbeschreibung ein passendes Objekt-Modell erstellt. Dadurch ist es möglich eine direkte Konvertierung in Objekte durchzuführen und ausschließlich mit diesen zu arbeiten. Diese Herausforderung ist gleichzeitig Voraussetzung für den im Kapitel 4.6 beschriebenen User Interface Generator.

Aus der Konvertierung ergeben sich Vorteile wie beispielsweise die Transparenz, da die Objekt- / Klassenstrukturen besser lesbar sind als codierte Zeichenfolgen. Des Weiteren wird die übermittelte Zeichenfolge bei der Konvertierung frühzeitig evaluiert und auf Korrektheit geprüft, da sie bei der Deserialisierung syntaktisch richtige sein muss. Beim Zugriff auf die enthaltenen Daten ergibt sich ein weiterer Vorteil, denn alle Inhalte sind im Gesamtmodell gleichzeitig verfügbar. Somit gestaltet sich die Verwendung des Modells sehr einfach. Ebenso können Vorteile der Vererbung ausgenutzt werden, wodurch Basisklassen nur einmalig implementiert werden müssen. Diese können zusätzlich um weitere Hilfsmethoden erweitert werden, um den Zugriff Inhalte zu erleichtern.

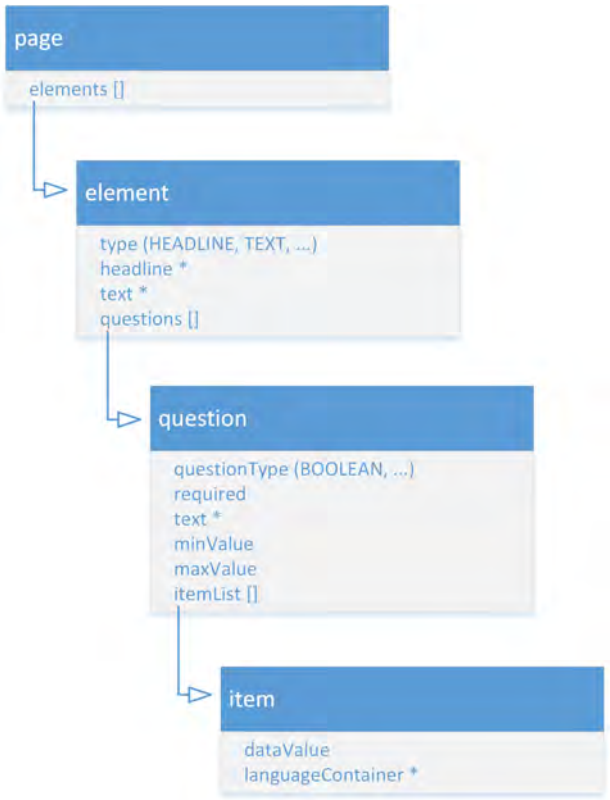


Abbildung 4.3.: Komprimierter Aufbau einer Seite

4.3.2. Ausführungslogik

Die Reihenfolge der dargestellten Fragebogenseiten ist in der zu realisierenden Anwendung nicht mehr dirrekt im Quellcode festgelegt, sondern wird von der Konfigurationsumgebung vorgegeben. Dazu modelliert der Konfigurator für jeden Fragebogen eine Ausführungslogik. Diese kann von einer Prozess Engine (siehe Kapitel 4.4) gelesen und interpretiert werden. Die eingesetzte leichtgewichtige Prozess Engine wurde im Rahmen einer Projektarbeit für diesen Anwendungsfall entwickelt und ist Teil des hier beschriebenen Konzepts und der Anwendung.

4.4. Prozess Engine

Die Prozess Engine dient als zentrale Kommunikationskomponente zwischen der eigentlichen Anwendung und der Datenbank. Von ihr werden folgende Funktionsgruppen aus Sicht der mobilen Anwendung bereit gestellt:

Verwalten von Fragebögen: Bietet Basisfunktionen in Bezug auf Fragebögen an, beispielsweise das Importieren, Löschen und Instanziieren von Fragebögen.

Verwalten von Instanzen eines Fragebogens: Für jeden gestarteten Fragebogen wird von der Prozess Engine eine neue Instanz zur Identifikation erzeugt. Diese Instanzen werden von der Prozess Engine in der Datenbank verwaltet und können über bereitgestellte Schnittstellenfunktionen bearbeitet werden.

Bereitstellung von Funktionen auf Knoten: Gefordert werden hier Funktionen zum Starten, Stoppen, Abschließen, Pausieren und Wiederaufnehmen von Knoten. Im Gesamtsystem repräsentiert ein Knoten eine Seite eines Fragebogens auf der Benutzeroberfläche und ist somit zentrales Objekt bei der dynamischen Generierung. Der Aufbau eines Fragebogens wurde bereits in Kapitel 4.3 beschrieben.

Verwalten der Antworten: Alle erhobenen Antworten werden über die Schnittstellenfunktionen der Prozess Engine in die Datenbank gespeichert, zu einem späteren Zeitpunkt wieder gelesen und an der Server übermittelt.

Steuern des Ablaufes eines Interviews: Wird die Anwendung aus Sicht der Prozess Engine betrachtet, ist diese lediglich eine Komponente zur Visualisierung einer einzelnen Seite eines Fragebogen. Genauer dargestellt wird dieser Sachverhalt im Kapitel 4.5.

4.5. Zusammenspiel der Komponenten

In diesem Kapitel wird die Kommunikation zwischen den einzelnen Komponenten der Anwendung beschrieben, um den Ablauf vom Anfragen eines Fragebogens bis zum

Zurücksenden darzustellen. Dabei wird im Speziellen auf die Kommunikation mit der Prozess Engine eingegangen.

4.5.1. Kommunikation der Komponenten

In der Abbildung 4.4 wird der komplette Ablauf vom Laden eines Fragebogens, über das Erheben von Daten, bis hin zum Rücksenden der Antworten schematisch in 11 Schritten dargestellt. Diese Schritte beschreiben die Eckpunkte des konzeptionellen Ablaufes der Anwendung in einer vereinfachten Form. Hierbei wird besonders Wert auf den systematischen Aufbau und die geschaffene Architektur gelegt. Die wichtigsten Aspekte der Abbildung 4.4 werden im Folgenden detaillierter erklärt.

Schritt 1 - Fragebögen laden: Da die Anwendung keine Funktion zum Erstellen eines Fragebogens bereit stellt, werden im Schritt 1 die Fragebögen vom Server auf das mobile Endgerät geladen. Dazu implementiert der Server, sowie das mobile Endgerät eine REST Schnittstelle, welche über eine WADL-Datei definiert ist. Unter Verwendung dieser Web Schnittstelle können Fragebögen auf das Endgerät transportiert werden. Ebenso enthält ein Fragebogen eine Ausführungslogik für die Prozess Engine, womit diese den Ablauf des Fragebogens steuert. Der detaillierte Aufbau einer Seite inklusive dessen Inhalt wurde im Kapitel 4.3 beschrieben.

Schritt 2 - Fragebögen speichern: Die im Schritt 1 heruntergeleadenen Fragebögen werden nun an die *Prozess Engine* weitergegeben und dort in der lokalen Datenbank persistiert. Das ist notwendig, um die Fragebögen ohne dauerhafte Verbindung zum Server ausführen zu können.

Schritt 3 - Fragebögen laden: Soll nun ein Fragebogen gestartet werden, muss dieser von der *Prozess Engine* angefragt werden. Die Prozess Engine liefert so Meta Informationen zum Fragebogen, die angezeigt werden.

Schritt 4 - Fragebögen anzeigen: Nachdem die Daten von der Prozess Engine erhalten wurden, visualisiert die *Startseite* dem Benutzer die verfügbaren Fragebögen.

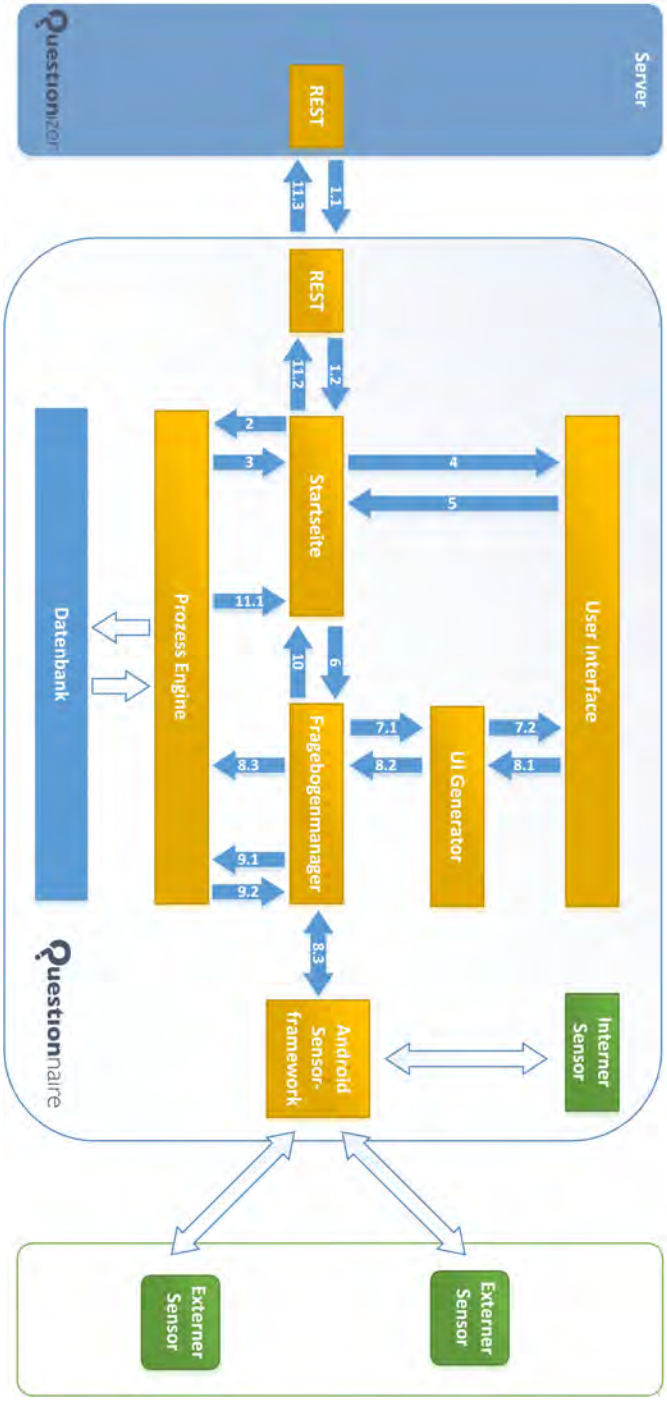


Abbildung 4.4.: Schematische Systemarchitektur

Zur besseren Übersicht werden diese in zwei Bereiche *Fragebögen* und *Pausierte Fragebogen* eingeteilt.

Schritt 5 & 6 - Fragebogen starten: Der Benutzer wählt den zu startenden Fragebogen in einer vorhandenen Sprache aus und startet diesen durch Betätigen des *Start* Buttons. Dadurch wird der *Fragebogenmanager* mit dem gewählten Fragebogen gestartet. In dem hier beschriebenen Ablauf wird nur auf das Starten von *neuen* Fragebögen eingegangen. In der späteren Verwendung der Anwendung besteht zusätzlich die Möglichkeit pausierte Fragebogen fortzuführen. Dabei wird letzendlich als zusätzlicher Übergabeparameter die Instanz ID übergeben. Ob eine Instanz eines Fragebogens pausiert ist, kann anhand des Zustandes erkannt werden.

Schritt 7 - Fragebogenseite anzeigen: Im *Fragebogenmanager* wird unter Verwendung der Schnittstellen zur Prozess Engine eine neue Instanz des Fragebogens angelegt und der zu startende Knoten angefragt. Nach Erhalt wird dieser gestartet und die dort enthaltene Seitenbeschreibung ausgelesen, welcher die Information für den dynamischen Seitenaufbau enthält. Auf Grundlage der Seitenbeschreibung generiert der *User Interface Generator* ein dazu passendes Layout und zeigt dieses auf der Benutzeroberfläche an. Eine detailliertere Beschreibung ist im Kapitel 4.6 enthalten.

Schritt 8 - Fragebogenseite beantworten: Nachdem der *User Interface Generator* die Oberfläche fertig erstellt hat, kann der Benutzer nun mit dem mobilen Endgerät interagieren und die Fragen beantworten. Jede so erhobene Antwort wird direkt an die *Prozess Engine* geschickt und in der Datenbank persistiert. Falls für die ganze Seite, oder nur für eine einzelne Frage zusätzliche Sensordaten verlangt werden, so findet an dieser Stelle eine Interaktion mit dem *Sensor Framework* statt. Dadurch ist es möglich die Antwort des Benutzers mit zusätzlichen Informationen wie beispielsweise Audioaufzeichnungen anzureichern. Unter Verwendung von externen Sensoren, welche über Bluetooth angebunden werden können, sind auch Vitalwerte (wie Puls oder Sauerstoffsättigung) als Sensordaten denkbar. Sind alle

Fragen beantwortet, kann der Benutzer durch Betätigen des *Weiter* Buttons die nächste Seite aufrufen.

Schritt 9 - Fragebogenseite wechseln: Durch betätigen des *Weiter* Buttons wird der Knoten abgeschlossen. Anschließend wird bei der Engine der nächste Knoten, also die nächste Seite im Fragebogen angefragt. Falls der Fragebogen noch weitere Seiten enthält, wird der *Fragebogenmanager* erneut aufgerufen. Dieser erhält dabei als Übergabeparameter die Fragebogen ID und die Instanz ID. Mit diesen beiden Informationen kann unter Verwendung von Schritt 7 die nächste Seite angezeigt werden. Ab hier können sich Schritt 7, 8 und 9 beliebig oft wiederholen. Falls bereits die letzte Seite beantwortet wurde, wird der Fragebogen abgeschlossen und durch die Prozess Engine persistiert.

Schritt 10 - Fragebogen abschliessen: Nach erfolgreichem Abschluss des Fragebogens wird wieder zur *Startseite* zurückgewechselt und die dort enthaltenen Daten aktualisiert. An dieser Stelle kann mit Schritt 5 ein weiterer Fragebogen ausgeführt oder Schritt 11 in die Administration gewechselt werden.

Schritt 11 - Fragebogen auswerten: Über einen Passwort-geschützten *Administrationsbereich*, können die erhobenen Daten von der Prozess Engine gelesen und über das vorher beschriebene Web Interface zum Server gesendet werden. Dort besteht die Möglichkeit, diese Daten durch Analysen strukturiert auszuwerten.

4.5.2. Steuerung durch die Engine

Einfach gesagt fungiert die Engine als Controller zur Steuerung der auf dem Konfigurator definierten Abläufe des Fragebogens. Die Abbildung 4.5 zeigt die Steuerung der Anwendung durch die Prozess Engine bei der Erstellung des generischen Layouts. Die wichtigste Funktion hierbei ist `getStartableNode`, durch deren Rückgabewert eine Liste von startbaren Knoten dem `User Interface Generator` mitgeteilt werden.

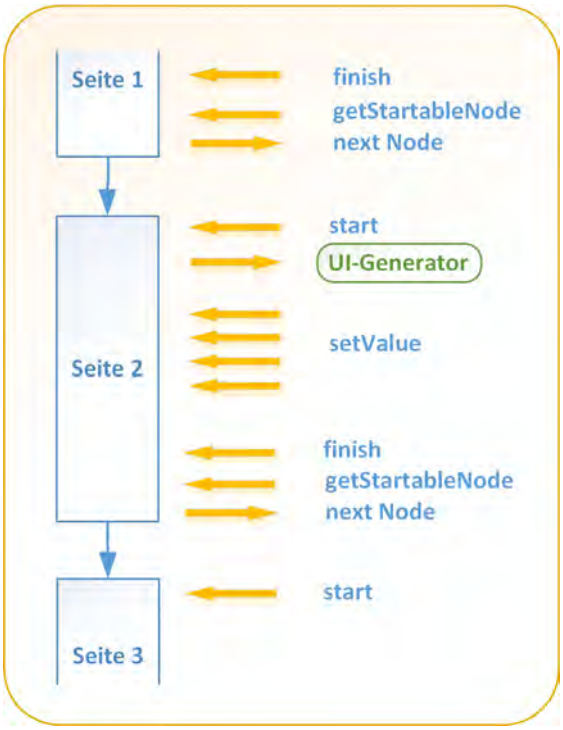


Abbildung 4.5.: Steuerung der Anwendung durch die Prozess Engine

4.6. User Interface Generator

Der User Interface Generator ist das Herzstück der Anwendung. Er ist dafür verantwortlich, die Objekt Repräsentation der JSON Zeichenfolge dem Benutzer mittels Kombination verschiedenster Basiselemente anzuzeigen. Die Realisierung und der Aufbau dieser Elemente wird im Kapitel 6 beschrieben.

Der User Interface Generator benötigt zum dynamischen Aufbau der Benutzeroberfläche die Seitendefiniton als Objektmodell (sieht Kapitel 4.3.1). Als wichtigste Attribute kristallisieren sich die Typen zur Unterscheidung der User Interface Elemente und die Texteressourcen zur Darstellung der jeweiligen Übersetzungen heraus. Dabei wird bei der dynamischen Generierung der Oberfläche zwischen den Element Typen (`type`, siehe Kapitel 4.3) unterschieden. Diese werden bei der Modellierung durch den Konfigurator zugeordnet. Abbildung 4.8 und 4.7 zeigen die Basiselemente der Typen.

Für Frageblöcke und Fragegruppen (`QUESTION`, `QUESTION_BLOCK`, `QUESTION_GROUP`) wird kein extra Basis-Element benötigt, da diese nur eine Kombination aus einzelnen Basis-Fragen-Elementen (sieht Abbildung 4.8) sind, welche nachfolgend unterschieden und gezeigt werden.

Abbildung 4.6.: Basiselement für Fragen

(a) HEADLINE (b) TEXT

Abbildung 4.7.: Basiselemente für Text

Bei allen Elementen des Typs `QUESTION` muss zusätzlich zwischen den Basis-Fragen-Elementen differenziert werden. Diese beschreiben die Art der Frage und sind am Attribut `questionType` erkennbar, welches jede Frage besitzt (`questionType`, siehe Kapitel 4.3). Abbildungen 4.8 a) bis h) zeigen die verfügbaren Basis-Fragen-Elemente mit den dazugehörigen Typen.

(a) Zweistellungsschalter - BOOLEAN

(b) Dreistellungsschalter mit neutraler Stellung - BOOLEAN_EXTENDED

(c) Mehrstellungsschalter- SLIDER

(d) Texteingabe - TEXT_INPUT

(e) Zifferneingabe - NUMBER_INPUT

(f) Datumseingabe - DATE_INPUT

(g) Bereichseingabe - RANGE_INPUT

(h) Einfachauswahl - SINGLE_CHOICE

Abbildung 4.8.: Basiselemente

4.7. Konzept der Editoren

Im Kapitel 4.6 wurden die Basiselemente für die verschiedensten Frage-Typen erläutert. Die meisten von ihnen werden in der dargelegten Arbeit zur `Darstellung` von Daten in geeigneter Form verwendent. Diesen gegenüber stehen `Editoren`, mit welchen die Bearbeitung der visualisierten Daten möglich ist.

Eine Ausnahme bildet die Gruppe von Interaktionselementen, welche gleichzeitig als Editor fungieren. Aus Sicht der Usability sind diese die effizientesten Bedienelemente, da der dargestellte Wert direkt (das heißt ohne zusätzliche Interaktion mit der Benutzeroberfläche) verändert werden kann. Dies ist in der Regel nur dann möglich, falls das Intervall der Datenwerte klein, diskret und beschränkt ist. In der hier dargelegten Arbeit nimmt beispielsweise ein Mehrstellungsschalter diese Rolle ein (siehe Abbildung 4.8 c)).

Im Folgenden werden alle Editoren dargestellt, welche in der vorliegenden Anwendung umgesetzt wurden, um Benutzereingaben zu verarbeiten:

Editor für Text: Der wohl meist bekannteste Editor ist die einfache Tastatur zur Eingabe und Änderung von Texten. In allen Betriebssystemen für mobile Endgeräte ist bereits eine Tastatur enthalten. Deshalb wurde beschlossen, kein eigener Editor für

Text-Eingaben zu entwerfen, sondern auf den systeminternen zurückzugreifen. Aus Sicht der Usability ist diese Entscheidung zu begrüßen, denn durch Verwendung von systemeigener Komponenten verhält sich die zu realisierende Anwendung dem Beutzer gegenüber erwartungskonform. Somit muss sich dieser Anwender nicht mit neuen Elementen beschäftigen. Abbildung 4.9 zeigt beispielhaft einen Editor für Text-Eingaben.

Abbildung 4.9.: Editor für Text-Eingabe (Android Framework)

Editor für Ziffern: Bei dem Editor für Ziffern gelten die gleichen Aussagen, welche bereits für den Editor zur Text-Eingabe aufgeführt wurde. Auch hier wird ein systemeigener Editor für Ziffern benutzt. Abbildung 4.10 zeigt beispielhaft einen Editor für Ziffern.

Abbildung 4.10.: Editor für Ziffern (Android Framework)

Editor für Datum: Der Editor für Datum ist ein neu entworfener und eigens für diese Anwendung umgesetzter Editor. Hierbei wurde darauf geachtet, dass es dem Benutzer leicht gemacht wird, ein entsprechendes Datum auszuwählen. Aus Sicht der Usability ist hierfür die Repräsentation durch einen Kalender passend. Abbildung 4.11 zeigt einen Editor für Datumswerte.

Abbildung 4.11.: Editor für Datum (neues Bedienelement)

Editor für Bereiche: Auch der Editor für die Bereichseingabe wurde neu entworfen und realisiert. Dabei wurde auf ein klare und einfache Darstellung von Werten innerhalb eines definierten Wertebereiches geachtet. Die Abbildung 4.12 zeigt den Editor für Bereichseingaben. Selektierte Werte werden dabei als gedrückte Buttons dargestellt.

Abbildung 4.12.: Editor für Bereiche (neues Bedienelement)

4.8. Multilingualität

In der funktionalen Anforderung 3.2.1 wurde die Mehrsprachigkeit der Anwendung bereits aufgeführt und die Gründe dafür erläutert (siehe Kapitel 3). Das zugrunde liegende Konzept wird hier diskutiert.

In der hier erstellten Anwendung ist es vorgesehen, dass Probanden einen Fragebogen selbst lesen und beantworten. Dabei ist nicht vorauszusetzen, dass die betreuende Person und die Probanden die selben Sprachkenntnisse besitzen. Des weiteren ist die Bearbeitung eines Fragebogens in der jeweiligen Muttersprache einfacher, da die Fragen besser verstanden werden. Aus diesen Gründen wird ein zweigeteiltes Sprachkonzept verwendet, um die Sprache der Anwendung und die des Fragebogens getrennt voneinander behandeln.

In Bezug auf die Anwendung bietet die *Startseite* eine Funktion zur Umschaltung der Anwendungssprache. Dabei ist zu beachten, dass die Sprache nach der Umschaltung live geändert wird. Die Ressourcen dafür müssen bereits in der erstellten Anwendung enthalten sein. Um die Anwendungssprache zu erweitern, muss eine neue Version der Anwendung mit den erweiterten Sprachressourcen erstellt werden.

Die Sprache des Fragebogens hingegen kann bei der Auswahl eines Fragebogens umgestellt werden. Alle unterstützten Sprachen sind jeweils im Fragebogen enthalten. Neue Sprachen können in der Konfiguratorumgebung beim modellieren eines Fragebogens hinzugefügt werden. Dazu muss anschließend der Fragebogen neu auf das Gerät übertragen werden. Jede Sprachressource wird dabei als `languageTextContainer` dargestellt (siehe Kapitel 4.3). Dieser Container enthält alle länderspezifische Übersetzungen, welche für einen einzelnen Text konfiguriert sind.

5

Walkthrough

In diesem Kapitel wird der konzeptionelle Ablauf vom Start der Anwendung, bis zum Ende eines dynamisch generierten Fragebogens in sinnvoll gekürzter Form dargestellt. Dabei wird der Fokus auf die verschiedenen Dialoge so wie wichtigsten Funktionen gelegt. Besondere Beachtung findet auch die Generierung des User Interfaces. In Abbildung 5.1 wird der schematische Ablauf der Activities mit dem Fokus des generischen Seitenaufbaus dargestellt.

Das Ziel diese Kapitels ist es, die Benutzeroberfläche vorzustellen und Transparenz für Basisprozesse der Anwendung zu schaffen.
Da die verschiedenen Sprachen am Ablauf der Anwendung nichts ändern und äquivalent sind, wird die Anwendung im Folgenden selber in englischer Sprache und die Fragebögen in deutscher Sprache angezeigt.

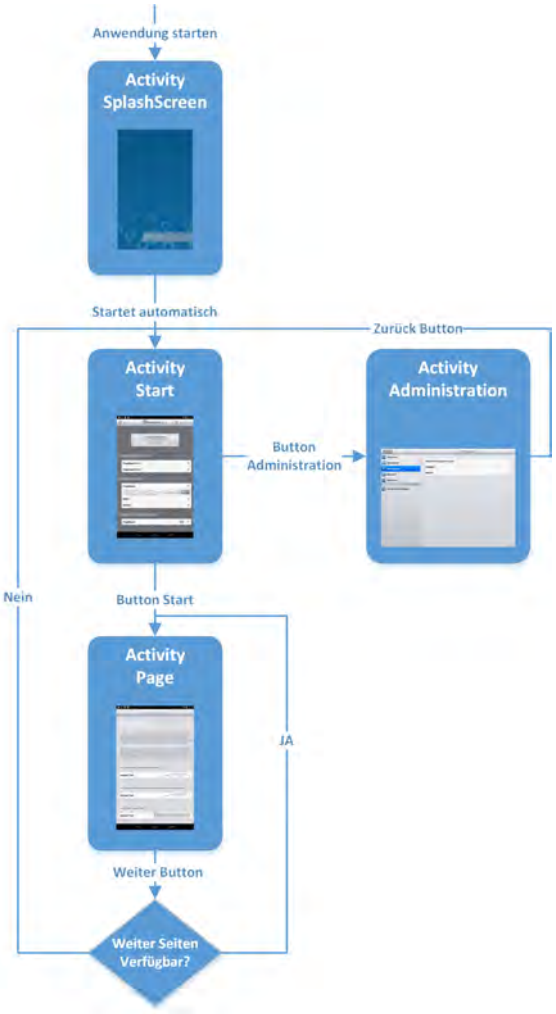


Abbildung 5.1.: Schematischer Ablauf der Activities

5.1. Start der Anwendung

Zu Beginn startet die betreuende Person die Anwendung über das Questionnaire Icon (Abbildung 5.2), woraufhin der Startbildschirm der Anwendung im Vollbildmodus angezeigt wird.

Abbildung 5.2.: Icon Questionnaire zum Starten der Anwendung

5.2. Startbildschirm

Der Startbildschirm (siehe Abbildung 5.3) bleibt so lange sichtbar, bis eine Instanz der Prozess Engine angelegt worden ist. Danach wird automatisch die ActivityStart gestartet. Aus Sicht des Benutzers bietet dieser Dialog keine nennenswerten Funktionen und muss somit nicht näher beschrieben werden.

Abbildung 5.3.: Aktivität SplashScreen

5.3. Startseite

Die Startseite stellt die wichtigesten Funktionen zur Durchführung von Interviews und Fragebögen dar. Aus diesem Grund ist er leicht verständlich, schlicht und einfach aufgebaut, damit jede für das System bestimmte Nutzergruppe mit der Oberfläche interagieren kann. Die Oberfläche gliedert sich in eine Kopfzeile zur Navigation und in zwei weitere Bereiche, in welchen die auf dem System befindlichen Fragebögen anzeigt werden.

In der Kopfzeile besteht die Möglichkeit, die Sprache der Anwendung umzuschalten (Abbildung 5.4), sowie in den passwortgeschützten Administrationsbereich zu wechseln zu wechseln.

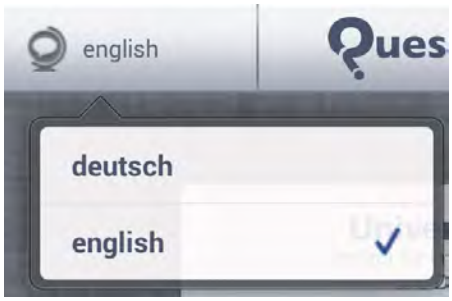


Abbildung 5.4.: Dialog zur Auswahl der Sprache

Die Bereiche für die Fragebögen unterteilen sich in:

- Fragebögen
- Pausierte Fragebögen

Jeder Eintrag in einem Bereich kann mit einem Klick auf den symbolisierten Pfeil an der rechten Seite auf- und zugeklappt werden. Im aufgeklappten Zustand wird ein zusätzlicher Beschreibungstext sichtbar, welcher den Fragebogen charakterisiert. Des Weiteren besteht dort die Möglichkeit die Sprache explizit für diesen Fragbogen umzustellen. Als Letztes findet man noch einen Button um den Fragebogen zu starten,

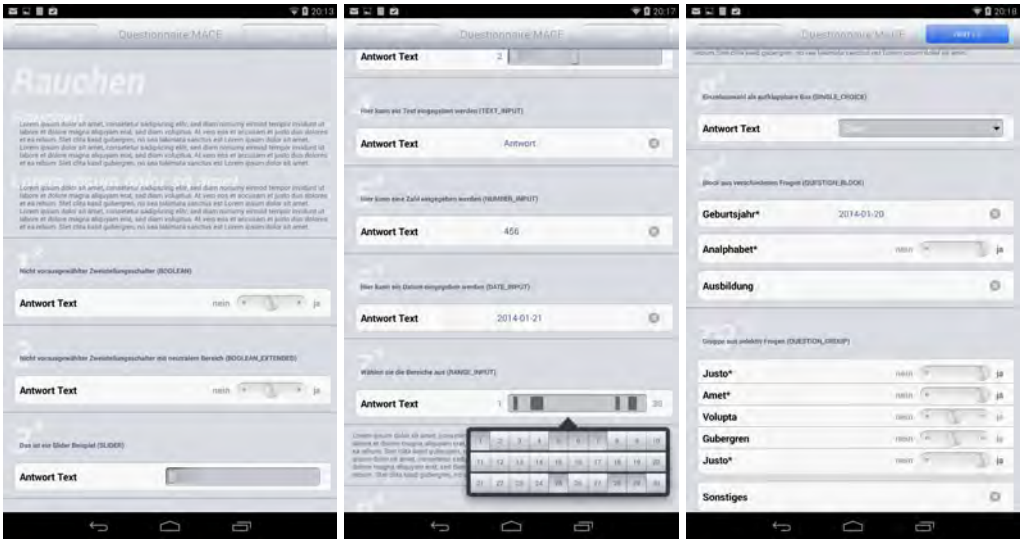
wodurch die Startseite aufgerufen wird.

Abbildung 5.5.: Aktivität Start

5.4. Fragebogenseite

Der Inhalt dieser Oberfläche, ist bis auf die Kopfzeile, komplett dynamisch generiert. In den Abbildungen 5.6 (a,b,c) sind die wesentlichen Elemente zu sehen, welche von dieser Anwendung unterstützt werden. Eine detailliertere Beschreibung der User Interface Elemente wurde bereits in Kapitel 3.2.9 diskutiert.

Die Kopfzeile beinhaltet einen *back* und einen *next* Button. Der *next* Button auf der rechten Seite hat initial die Eigenschaft nicht gedrückt werden zu können. Dies (siehe Kapitel 3.2.10) lässt sich aus den Anforderungen für den *Weiter* Button ableiten. Sobald für alle zwingend notwendigen Fragen eine Antwort vorhanden ist, wird der Button



(a) Fragebogenseite (1) (b) Fragebogenseite (2) (c) Fragebogenseite (3)

Abbildung 5.6.: Fragebogenseite

bedienbar. Beim Klick auf den *Weiter* Button werden zwei Fälle berücksichtigt, welche auf der Abbildung 5.1 dargestellt sind.

Falls es eine weitere Seite für diesen Fragebogen gibt, welche noch nicht beantwortet ist, so wird die Fragebogenseite von dieser Stelle aus erneut aufgerufen und zum Anzeigen der nächsten Seite aufgefordert. Falls alle Fragen beantwortet sind, so wird die ActivityStart gestartet und wir befinden uns wieder am Ausgangspunkt.

6

Implementierung

In den bisherigen Kapiteln wurden Anforderungen an die Anwendung festgehalten und Konzepte dafür vorgestellt, jedoch ohne detailiert die Implementierung zu betrachten. Somit können die bisher erarbeiteten Punkte ebenso als Basis für Realisierungen auf anderen Plattformen gesehen werden.

Die folgenden Kapitel bieten somit einen Einstieg in die konkrete Realisierung auf Basis des gewählten Android Frameworks. Zur besseren Veranschaulichung wird an einigen Stellen direkt Bezug auf Quellcode genommen, um ein tieferes Verständnis für die Umsetzung zu schaffen.
Dem Kapitel vorweg genommen ist der Einsatz der Programmiersprach JAVA, denn native Anwendungen für das Android Betriebssystem von Google werden in dieser Sprache realisiert. Dazu bietet Google das eigenständige SDK *ADT*, an, welches die Sprache JAVA um Android spezifische Bibliotheken erweitert. Damit wird der Grundstein

gelegt um Anwendungen für Android zu entwickeln. Ferner sind wichtige Hilfsprogramme enthalten, welche beispielsweise ein Gerät simulieren.
Des Weiteren wurden bei der Realisierung der hier vorgelegten Arbeit die klassischen Entwicklungsmodelle wie beispielsweise Wasserfall- oder V-Modell außen vor gelassen. Diese weisen Nachteile bezüglich der vorgelagerten Dokumentation auf (beispielsweise die Erstellung eines Lastenheftes) und verzögern dadurch den Projektstart [agi14]. Deshalb wurden beim Start des Projektes keine Schnittstellendefinitionen oder Funktionsdiagramme erstellt und in Lasten-/Pflichtenheft niedergeschrieben, denn dafür ist die Gesamtlaufzeit zu eng bemessen. Vielmehr wurde über ein generelles Konzept für das Gesamt-System nachgedacht, ohne dabei zu sehr ins Detail zu gehen.

6.1. Android

In diesem Abschnitt werden spezielle Eigenschaften des Android Frameworks vorgestellt, welche bei der Umsetzung eine wichtige Rolle gespielt haben und bei der Weiterentwicklung der hier vorliegenden Arbeit beachtet werden sollten.

6.1.1. 9 Patch Tool

Bei der Verwendung von Hintergrundbildern muss aufgepasst werden, dass diese nicht verzerrt angezeigt werden. Das kann beispielsweise durch Verwendung verschiedenster Bildschirmauflösungen oder Seitenverhältnisse hervorgerufen werden, welche der großen Geräte- und Herstellervielfalt von Android geschuldet ist. Um der beschriebenen Problematik aus dem Weg zu gehen, bietet die *Android Developer Tools* Installation ein Programm namens 9Patch. Damit kann für Grafiken angegeben werden, welche Teile skaliert werden können und welche immer statisch bleiben müssen [9Pa14]. In der Abbildung 6.1 wird dargestellt, wie sich die Darstellung unter Verwendung von 9Patch ändert.

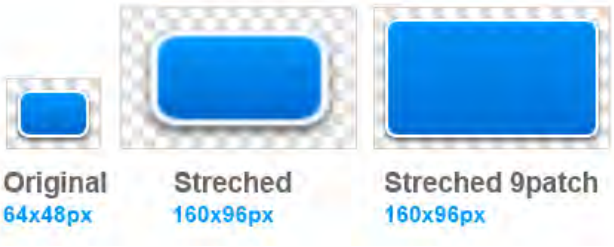


Abbildung 6.1.: Beispiel für die Verwendung von 9Patch [9Pa14]

6.1.2. Ressourcen

Bei einem Android Projekt sind Ressourcen unterhalb des dafür vorgesehenen Ordners `/res` zu finden. Dabei werden mittels der Ordnerstruktur die Typen von Ressourcen unterschieden. Nachfolgend werden die Typen Bild, Layout und Text diskutiert.

Bildressourcen: Es gibt mehrere Möglichkeiten Bildressourcen in einem Android Projekt abzulegen. Ein Ort dafür ist der Ordner `res/drawable`. Alle dort enthaltenen Bilder werden vom Android Framework selbst auf die jeweilige Auflösung und das Verhältnis des Bildschirmes unterschiedlicher Geräte skaliert. Dabei kann es passieren, daß einige Bilder nicht wie erwartet dargestellt werden. Deshalb kann für jede von Android unterstützte Auflösung ein Bild angefertigt und in den dafür vorgesehenen Ordner abgelegt werden. Dazu können die Ordner `res/drawable-ldpi` (120 dpi), `res/drawable-mdpi` (160 dpi), `res/drawable-hdpi` (240 dpi), `res/drawable-xhdpi` (320 dpi), `res/drawable-xxhdpi` (480 dpi), und `res/drawable-xxxhdpi` (640 dpi) verwendet werden [And14] [Kü12].

Layouts: Um Layouts für verschiedene Bildschirmgrößend und Bildschirmauflösungen zu Verwalten kann ähnlich vorgegangen werden, wie bereits im Kapitel 6.1.2 erklärt wurde. Dabei ist zu beachten , dass das allgemeine Layout im Ordner `res/layout` liegt. Die spezifische Layouts dagegen müssen je nach Bildschirmgröße in den Ordnern `res/layout-small` (2 bis 3,2 Zoll), `res/layout-normal` (3,2 bis 4 Zoll), `res/layout-large` (4 bis 7 Zoll) und `res/layout-xlarge` (7 bis 10 Zoll) abgelegt werden [And14] [Kü12].

Textressourcen: Wie bereits im Kapitel 4.8 diskutiert, ist das Sprachkonzept der realisierten Anwendung zweigeteilt. Zum einen in die Sprachen des Fragebogens und zum anderen in die der Anwendung. Alle für die Anwendung notwendigen Texte befinden sich dabei in der Datei `res/values/strings.xml`. Diese Datei wird vom Android Framework als Standardsprache verwendet. Um eine weitere Sprache in der Anwendung zu verwenden muss die Datei `strings.xml` in einen neuen Ordner kopiert (beispielsweise `res/values-de/` für Deutsch) und anschließend in die jeweilige Sprache übersetzt werden. Der Codeausschnittes C.1 zeigt, wie in einem Android Projekt die Anwendungssprache geändert wird.

```
1 public void changeLanguage(String language) {
2     // Konfigurationsobjekt erstellen und neue Sprache setzen
3   android.content.res.Configuration config = new android.content.res.
        Configuration();
4   config.locale = mLocaleLanguage;
5   Resources res = this.mLocalActitity.getBaseContext().getResources();
6   DisplayMetrics dm = res.getDisplayMetrics();
7   // Sprache aendern
8   res.updateConfiguration(config, dm);
9 }
```

Listing 6.1: Ändern der Anwendungssprache

6.2. Multilingualität

Im Kapitel 6.1.2 wurde bereits erklärt, wie neue Sprachen für die Anwendung hinzugefügt werden können. Für den Fragebogen selber werden Textressourcen über `languageTextContainer` (siehe Kapitel 4.3) abgebildet. Der Codeausschnitt 6.2 zeigt beispielhaft den Aufbau eines `languageTextContainers` für die Textressource *Rauchen* in JSON. `languageCode` ist dabei der verwendete länderspezifische Code nach ISO 639 und `languageValue` die dazugehörige Übersetzung.

```
languageTextContainer: [
  {
    "languageCode": "de-de",
    "languageValue": "Rauchen"
  },{
    "languageCode": "en-us",
    "languageValue": "Smoking"
  }
]
```

Listing 6.2: Aufbau eines LanguageTextContainers

6.3. Button zum Starten von Fragebögen

Für jeden Fragebogen, welcher von der Prozess Engine angefragt wird, generiert die *Activity Start* einen Start Button zum Starten des Fragebogens. Dabei ist besonders zu beachten, dass es für alle Buttons eine gemeinsame `OnClick` Methode gibt. Um die Buttons zu unterscheiden wird dem `Tag` des Buttons beim Erstellen ein DatenContainer gesetzt. Dieser enthält Informationen über das Template, die Instanz und der selektierten gewählten für den Fragebogen.

Beim Betätigen eines Buttons wird also die zentrale Methode `onClick` aufgerufen, der `Tag` des Button ausgelesen und der Fragebogen gestartet. Dabei werden als Übergabeparameter die Inhalte des im Button Tag entaltenen DatenContainers übergeben. Anhand des folgenden Codeausschnittes 6.3 lässt sich dieses Vorgehen leicht nachvollziehen.

```
@Override
public void onClick(View v) {
    // Button Tag des gedrueckten Buttons wird gelesen
  StartButtonDataContainer tag = (StartButtonDataContainer) v.getTag();

  Intent intent = getBaseContext().getPackageManager().
      getLaunchIntentForPackage(
              getBaseContext().getPackageName());
  intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
```

```
9
10    // Uebergabeparameter werden anhand des Tags gesetzt
11    intent.setClass(ActivityStart.this, ActivityPageManager.class);
12    intent.putExtra("language", tag.getSelectedLanguage().languageCode);
13    intent.putExtra("headline", tag.getHeadline());
14    intent.putExtra("templateId", tag.getUUID() != null ? tag.getUUID()
15        .toString() : null);
16    intent.putExtra("instanceId", tag.getInstanceID() != null ? tag
17        .getInstanceID().toString() : null);
18
19    // ActivityPageManager wird gestartet
20    this.startActivity(intent);
21 }
```

Listing 6.3: Zentrale onClick Methode für Start Buttons

6.4. Starten eines Fragebogens

Nach dem Start der Activity `ActivityPageManager` wird die Methode `startInstance` der Prozess Engine aufgerufen, wodurch anhand des Templates eine Instanz erstellt wird. Anschliessend wird auf dieser Instanz die Funktion `getStartableNode` aufgerufen, um einen startbaren Knoten, also eine Seite im Fragebogen, zu bekommen. Wie bereits in Kapitel 4.3 beschrieben, enthält jeder Knoten den Inhalt einer Seite in JSON. Um später dem User Interface Generator die nötigen Daten bereitstellen zu können, werden diese aus dem zu startenden Knoten ausgelesen und in ein repräsentatives Objektmodell gewandelt (siehe Kapitel 4.3.1). Als letzter Schritt wird der Knoten schlussendlich gestartet.

Die Methode `setContentData`, welche im Codeausschnitt 6.4 enthalten ist, zeigt den Ablauf vom Erstellen einer Instanz eines Fragebogens, bis zum Starten eines Knotens. Hierbei wurden zwei Teilaufgaben in separate Methoden ausgelagert. Zum Ersten die Methode `getNextNode`, welche von einer Instanz den nächsten zu startenden Knoten ermittelt. Zum Zweiten die Methode `getPageContentFromNode`, welche von einem Knoten die Informationen über die enthaltenen Seite ausliest und als Objektstruktur zurückgibt [Jav14].

```
@Override
private Boolean setContentData() {
  // Hier wird eine Instanz angelegt (falls nicht vorhanden)
  if (this.mInstanceID == null) {
    try {
      // Mit der Methode startInstance der Engine wird eine Instanz angelegt.
      String instanceString = this.mExecutioner
          .startInstance(this.mTemplateID.toString());
      InstanceDTO instanceDTO = null;

      // Wandeln der Instanz von String in ein Objekt
      instanceDTO = (InstanceDTO) XmlInterface.deserializeToObject(
          InstanceDTO.class, instanceString);
      this.mInstanceID = instanceDTO.getInstanceId();

    } catch (Exception e) {
      //
    }
  }

  // Hier wird der startbare Knoten ermittelt und gestartet
  NodeDTO activeNode;
  try {
    // startbaren Knoten holen
    activeNode = this.getNextNode(this.mInstanceID);
    this.mNodeID = activeNode.getNodeID();

    // Knoten Inahlt auslesen und in Objekt wandeln
    this.mContentModel = getPageContentFromNode(activeNode);

    // Starten des Knotens
    this.mExecutioner.startNode(String.valueOf(this.mInstanceID),
        this.mNodeID);
  } catch (Exception e) {
    //
  }
  return true;
}

private NodeDTO getNextNode(UUID instanceId) throws Exception {
  // getStarableNode gibt startbare Knotenliste zurueck
  String startableNodesString = this.mExecutioner
      .getStartableNode(instanceId.toString());

```

```
45    // wandeln der Liste in eine Objektliste
46    List<NodeDTO> startableNodes = ((NodesDTO) XmlInterface
47        .deserializeToObject(NodesDTO.class, startableNodesString))
48        .getNodeList();
49
50    if (startableNodes == null || startableNodes.size() == 0) {
51      return null;
52    }
53    return startableNodes.get(0);
54  }
```

Listing 6.4: Vom Erstellen einer Instanz bis zum Starten eines Knotens

6.5. Client Modell

Im Kapitel 4.3.1 wurde bereits erläutert, warum in der Anwendung eine interne Repräsentation der in JSON codierten Seitenbeschreibung notwendig ist. Die Abbildung 6.2 zeigt das Schema des auf dem mobilen Endgerät verwendeten Modells in UML Notation. Die Struktur des Modells ist gleich aufgebaut, wie bereits im Kapitel 4.3 beschrieben wurde. Dadurch wird sichergesetllt, dass die Konvertierung der Seitenbeschreibung automatisiert durchgeführt werden kann. Der wichtigste Vorteil dabei ist die frühzeitige Validierung durch die Konvertierung in das Objekt-Modell. In der Anwendung wird ab diesem Zeitpunkt ausschließlich auf dem hier vorgestellten Modell weitergearbeitet. Hierbei ist zu beachten, dass Erweiterungen des Konfigurator Modells auch Erweiterungen des Modells auf der mobilen Anwendung nach sich ziehen, um diese Modelle konsistent zu halten. Ein Objekt-Modell repräsentiert dabei eine Fragebogenseite und enthält alle Informationen, welche für den User Interface Generator notwendig sind, um die Benutzeroberfläche dynamisch aufzubauen. Es ist Voraussetzung für die Erstellung der Oberfläche und wird für jede Fragenbogenseite neu erstellt.

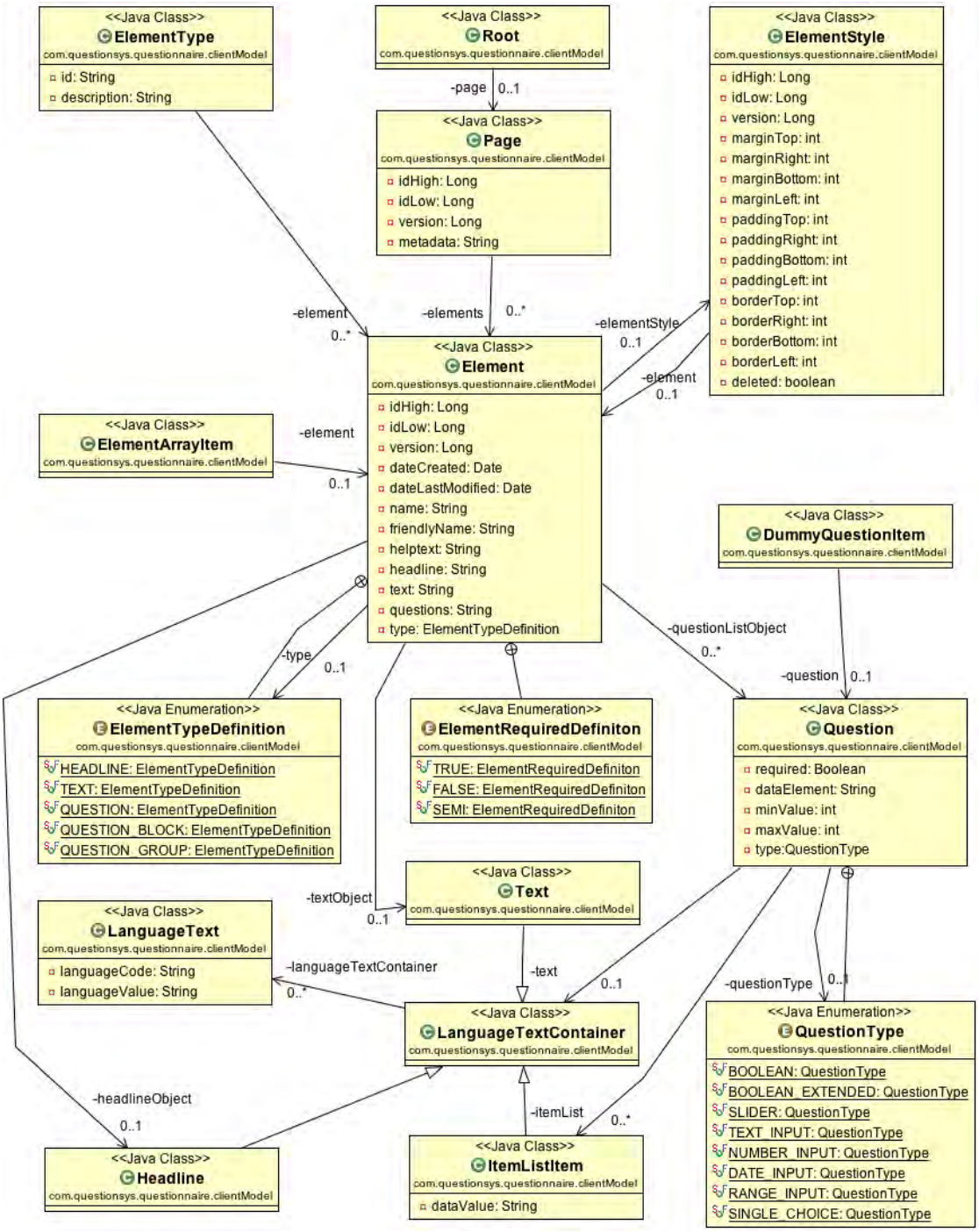


Abbildung 6.2.: Darstellung des Client Modells in UML Notation (Erstellt unter Verwendung von [Obj14])

Im Wesentlichen besteht das Modell aus den drei Grundelementen `Page`, `Element` und `Question`. Dabei kann das `Page` Objekt beliebig viele `Element` Objekte enthalten, welche Teilbereiche auf der Benutzeroberfläche beschreiben. Die `Element` Objekte werden anhand des Attributs `type` unterschieden, welches die Werte der Aufzählung `ElementTypeDefinition` annehmen kann. In Abhängigkeit des Typs sind die Attribute `headline`, `text` und `questions` mit Werten gefüllt. `headline` und `text` sind dabei vom Typ `languageTextContainer` und enthalten Textressourcen, welche zum Anzeigen auf der Benutzeroberfläche notwendig sind. Falls eine Frage angezeigt werden soll, so enthält die Liste `questions` Einträge vom Typ `Question`. Jede `Question` repräsentiert dabei eine Frage auf der Benutzeroberfläche. Diese werden über das Attribut `type` unterschieden, welches die Werte der Aufzählung `QuestionType` annehmen kann. Die darzustellenden Texte sind dabei in der Liste `dataElements` enthalten, wessen Einträge von der Klasse `languageTextContainer` ableiten. Schlussendlich legt das Attribute `required` fest, ob die Frage beantwortet werden muss und übersprungen werden kann.

6.6. User Interface Generator

Die konzeptionelle Beschreibung des User Interface Generators wurde bereits in Kapitel 4.6 diskutiert. Dort wurde im Detail aufgelistet, welche Oberflächenelemente unterstützt und nach welchen Kriterien diese unterschieden werden. Dieses Kapitel fokussiert auf die Implementierung der Oberflächenelemente. Dazu ist einleitend zu erwähnen, dass die Fragen über ihr `type` Attribut unterschieden werden (beispielhaft `type:RANGE_INPUT`). Für jeden definierten `type` wird in der Anwendung eine Oberklasse bereit gestellt, welche von der Basisklasse `QuestionListItem` ableitet. Im eben genannten Beispiel `RANGE_INPUT` wäre die dazugehörige Klasse `QuestionListItem_RangeInput`. Die Basisklasse wird im Kapitel 6.6 unter Verwendung von Codeausschnitten genauer erklärt.

Die Abbildung 6.3 zeigt im oberen Bereich die drei Elemente, zwischen denen anhand des Attributs `type` unterschieden wird. Die Elemente *Text* und *Headline* werden an dieser Stelle nicht weiter erklärt, da deren Aufbau sehr einfach ist. Ein *Frage* Element

jedoch setzt sich aus mehreren Einzelteilen zusammen, welche auf der Abbildung mit den Markierungen 1-4 gekennzeichnet sind.

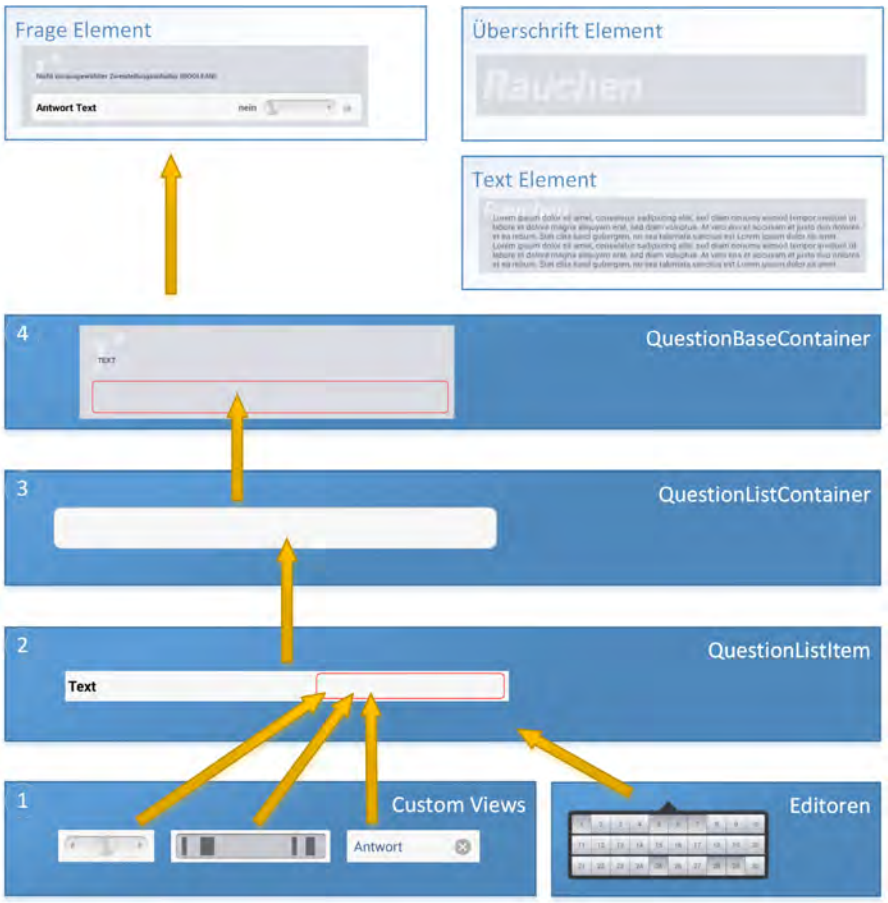


Abbildung 6.3.: Aufbau der Oberflächenelemente

Markierung 1: Für die hier dargelegte Arbeit wurden mehrere Interaktionselemente selbst gestaltet und realisiert (3.2.9), welche hier als `CustomViews` bezeichnet werden. In der Abbildung 6.3 sind exemplarisch ein Zweistellungsschalter, eine Bereichsauswahl und ein Texteingabeelement dargestellt. Diese `CustomViews` bilden die Grundelemente einer Frage. Hierbei ist zu beachten, dass nicht alle verwendeten Oberflächenelemente selbst realisiert wurden. Es können somit auch Standard Android Elemente als Grundelemente einer Frage verwendet werden.

Markierung 2: Die `CustomViews` oder bereits bestehende Android Elemente werden in ein `QuestionListItem` eingesetzt, welches die Basisklasse für alle Fragen darstellt. Dieser *Container* beinhaltet eine `TextView` an der linken und einen `LinearLayout` Container an der rechten Seite. In diesen zusätzlichen Container (in der Abbildung 6.3 rot umrandet) werden die unter der Markierung 1 genannten Interaktionselemente platziert. Falls das verwendete Grundelement selbst nicht editierbar ist, wird in diesem Schritt festgelegt, welcher Editor für das Grundelement verwendet werden soll. Dieser erscheint sobald das Interaktionselement auf der Benutzeroberfläche markiert wird. In der Abbildung 6.3 ist beispielhaft ein Editor für die Bereichsauswahl dargegstellt. Weitere Editoren zur Eingabe der Daten wurden bereits im Kapitel 4.7 beschrieben.

Markierung 3: In diesem Schritt wird ein `QuestionListItem` in einen `QuestionList Container` gesetzt. Bei der Darstellung von *Fragen Gruppen* (siehe Kapitel 3.2.9) enthält der genannte Container mehrere `QuestionListItem` Elemente. Der `QuestionListContainer` beinhaltet keine weitere Logik und wird nur für die strukturierte Darstellung verwendet.

Markierung 4: Letzendlich bietet ein `QuestionBaseContainer`, welcher das Basiselement für Fragen darstetllt, einen `LinearLayout` Container (in der Abbildung 6.3 rot umrandet). Zu diesem Container werden alle `QuestionListContainer` dieser Frage hinzugefügt (beispielsweise im Falle eines Fragenblocks werden mehrere `QuestionList Container` benötigt). Für die Darstellung der Fragennummer und des Fragentextes enthält der `QuestionBaseContainer` zwei `TextViews`.

QuestionListItem als Basisklasse

Die Klasse `QuestionListItem` dient als Basisklasse für alle Interaktionelemente vom `type Question` und enthält Hilfsfunktionen, sowie abstrakte Methoden. Die abstrakten Methoden müssen stets von der Oberklasse implementiert werden. Dadurch wird sichergestellt, dass bestimmte Schnittstellenmethoden immer vorhanden sind. Diese können nicht global für alle Oberklassen implementiert werden, da der enthaltenen Wert der

Antworten und dessen Typ (beispielsweise Integer, Boolean oder String) unterschiedlich sein kann. Im Folgenden werden die drei wichtigsten abstrakten Methoden erläutert:

`requiredFlagComplianceCheck`: In dieser Methode wird anhand des Attributes `required` einer Frage geprüft, ob diese zwingend beantwortet werden muss. Falls nicht, so gibt die Methode ein `true` zurück. Im anderen Fall wird zusätzlich geprüft, ob bereits eine Antwort vorliegt. Dabei wird `true` zurück gegeben, falls die Frage beantwortet ist, ansonsten `false`.

`getValue`: Diese Methode gibt den Wert der Antwort als String zurück. Dabei ist zu beachten, dass ein Wert einer Antwort unterschiedlich repräsensiert werden kann. Eine Bereichsauswahl liefert beispielsweise eine Liste von Werten und ein Zweistellungsschalter einen booleschen Wert. Aufgabe der Methode ist es, die erhaltenen *Rohdaten* von dem Interaktionselement in eine geeignete String Repräsentation zu bringen. Dazu werden die *Rohwerte* auf die im Objekt-Modell definierten Rückgabewerte (`dataValues`) gemappt.Es wird also bereits in der Konfiguratorumgebung festgelegt, welche Rückgabewerte eine Frage haben kann.

`setValue`: Hier wird die *Rückrichtung* der eben beschriebenen Methode definiert. Übergeben wird ein String, welcher einem im Objekt-Modell definierten Rückgabewert (`dataValue`) entspricht. Dieser muss auf einen *Rohwert* für das Interaktionelement gemappt und anschliessend visualisiert werden.

Damit Wertänderungen aus den Oberklassen registriert werden können, implementiert die Basisklasse ein `OnValueChanged` Event. Dieses Event muss bei jeder Änderung der Antwort vom jeweiligen Interaktionselement aufgerufen werden. Als Übergabeparameter beinhaltet dieser Aufruf die ID der Frage (`dataElementID`) und den Wert der Antwort (`value`). Um jede erhobene Antwort zu persistieren, wird darauf hin die Funktion `setValueForDataElementByName` der Prozess Engine mit den erhaltenen Werten aufgerufen.
Zur Vollständigkeit zeigt das UML Diagramm in Abbildung 6.4 alle von der Basisklasse `QuestionList Item` abgeleiteten Oberklassen welche in der hier dargelegten Arbeit verwendet werden.

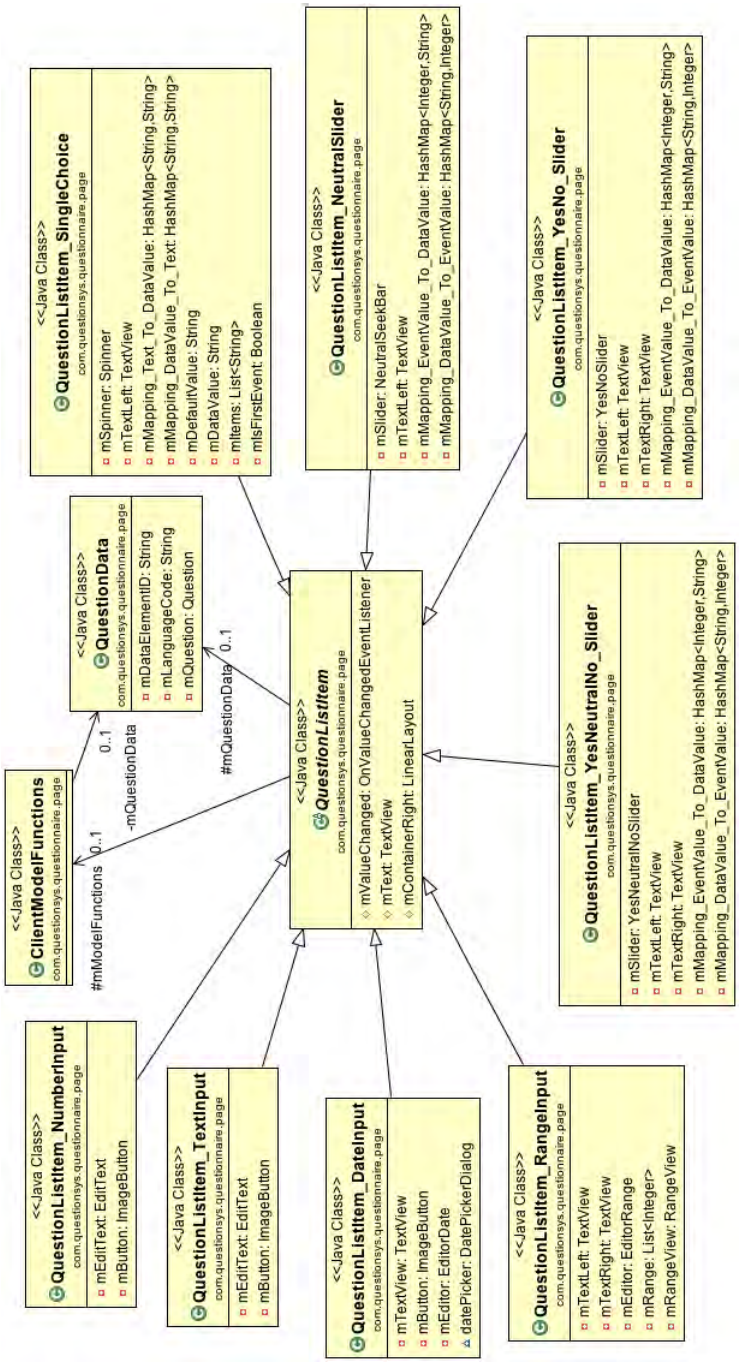


Abbildung 6.4.: Darstellung der Oberklassen zur Basisklasse QuestionListItem in UML Notation (Erstellt unter Verwendung von [Obj14])

6.7. ActivityPageManager

In der realisierten Anwendung ist die Aufgabe des `ActivityPageManager` das Anzeigen einer einzelnen Fragebogenseite und das Verwalten der erhobenen Daten. Die dazu gehörige Benutzeroberfläche wird dabei vom User Interface Generator (siehe Kapitel 4.6) erstellt.

Die Besonderheit, warum diese Activity als Manager deklariert wird, ist die zentrale Methode `onEvent`, welche im Codeausschnitt 6.5 gelistet ist. Diese Methode wird von allen `QuestionListItems` (siehe Kapitel 6.6) aufgerufen, da diese bei der Erstellung der Interaktionselemente für den `OnValueChanged` Event referenziert wurde. Der Manager hat beim Aufruf dieser Methode die Aufgabe den erhaltenen Wert an die Prozess Engine weiterzuleiten, die diesen anschließend speichert. Mit der Methode`setValueForDataElementByName` kann der erhaltene Wert gespeichert werden.

Als Zweites wird bei jedem Aufruf dieser Methode die Bedienbarkeit des Buttons zum Wechseln auf die nächste Fragebogenseite kontrolliert. Dazu stellt der Manager an jedes Objekt des Basistyps `QuestionListItem` die Anfrage, ob bereits eine Antwort vorliegt. Dazu wurde die abstrakte Methode `requiredFlagComplianceCheck` (siehe Kapitel 6.6) eingeführt. Falls alle Fragen unter Betrachtung des `required` Attributes beantwortet sind, so wird der Button für den Wechsel auf die nächste Fragebogenseite aktiviert, und der Anwender kann umblättern.

```
1  public void onEvent(String dataElementID, String value) {
2    // Schreibe neuen Wert in die Datenbank
3    // mExecutioner ist dabei die Prozess Engine
4    try {
5      this.mExecutioner.setValueForDataElement(
6          this.mInstanceID.toString(), this.mNodeID, dataElementID,
7          value == null ? "" : value);
8    } catch (Exception e) {
9      //
10   }
11
12   // Wechsel die Zustand des Weiter Buttons auf bedienbar, falls
```

```
13    // alle (required) Fragen beantwortet sind
14     for (QuestionListItem questionItem : this.mUiLayouter
15        .getQuestionItemList()) {
16      if (!questionItem.requiredFlagComplianceCheck()) {
17        mButtonNext.setEnabled(false);
18        return;
19      }
20    }
21    mButtonNext.setEnabled(true);
22  }
```

Listing 6.5: onEvent Methode der ActivityPageManager

6.8. Button zur Wechsel der Fragebogenseite

Mit dem *Weiter Button* kann auf eine neue Fragebogenseite gewechselt werden. Der Button ist allerdings erst dann bedienbar, wenn für alle zwingend notwendigen Fragen eine Antwort vorliegt. Dieser Sachverhalt wurde bereits in Kapitel 6.7 beschrieben.

Beim Betätigen dieses Buttons muss zwischen zwei Punkten differenziert werden. Bei beiden Punkten wird jedoch als erster Schritt die Instanz des Knotens auf den Status `finished` (siehe Kapitel 6.6) gesetzt. Es wird zwischen folgenden Fällen unterschieden:

- *Es existieren noch weitere unbeantwortete Fragenbogenseiten:* In diesem Fall wird die `ActivityPageManager` erneut gestartet. Dabei werden Template ID und Instanze ID des der aktuellen Seite übergeben. In der neu erzeugten `Activity PageManager` wird dann anhand der schon vorhandenen Instanz ID der nächste startbare Knoten bei der Prozess Engine angefragt. Es wird also keine neue Instanz erzeugt (was ein Neustart des Fragebogens wäre) sondern mit der nächsten Seite fortgefahren.
- *Die letzte Fragenbogenseiten wurde beantwortet:* Falls dies der Fall ist, so wird die aktuelle Activity beendet und auf die Startseite (`ActivityStart`) gewechselt. Damit mit dem *zurück Button* des Betriebssystems nicht auf die letzte Fragebogenseite gewechselt werden kann, wird zusätzlich die Methode `finish` aufgerufen.

Dadurch wird die Instanz der Activity vom Activity-Stack des Betriebssystems gelöscht.

```
1  public void onClick(View v) {
2    if (v == this.mButtonNext) {
3      try {
4        // Aktuellen Knoten auf den Status finished setzen
5        // mExecutioner ist dabei die Prozess Engine
6        this.mExecutioner.finishNode(this.mInstanceID.toString(),this.mNodeID);
7
8        if (this.getNextNode(this.mInstanceID) == null) {
9          // Aktuelle Activity verwerfen
10         super.onDestroy();
11
12       } else {
13         Intent intent = getBaseContext().getPackageManager()
14             .getLaunchIntentForPackage(
15                 getBaseContext().getPackageName());
16         intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
17
18         // Uebergabeparameter fuellen
19         intent.setClass(ActivityPageManager.this, ActivityPageManager.class);
20         intent.putExtra("language", this.mQuestionLanguageCode);
21         intent.putExtra("headline", this.mHeadline);
22         intent.putExtra("templateId", this.mTemplateID.toString());
23         intent.putExtra("instanceId", this.mInstanceID.toString());
24         intent.putExtra("questionCount",
25             this.mUiLayouter.getQuestionCount());
26         // Die ActivityPageManager erneut starten
27         this.startActivity(intent);
28       }
29
30       // Die aktuelle Aktivity Instanz vom Acitvity Stack loeschen.
31       // Damit ist sie nicht mehr ueber den zurueck Button aufrufbar
32       this.finish();
33     } catch (Exception e) {
34       //
35     }
36   }
37 }
```

Listing 6.6: onClick Methode der ActivityPageManager

6.9. Prozess Engine

In diesem Kapitel werden wichtige Aspekte in Bezug auf die Prozess Engine beschrieben. Wie die Engine die Steuerung der Anwendung übernimmt wurde bereits in Kaptitel 4.5.2 beschrieben.
Im Anhang B befindet sich die komplette Auflistung der geforderten Schnittstellen zur Prozess Engine.

6.9.1. Instanziierung der Prozess Engine

Die Prozess Engine ist nach dem *Singleton Pattern* [KE13] implementiert. Das bedeutet, dass maximal eine Instanz der Engine erzeugt werden kann. Der Grund für die Wahl eines Singletons ist zum einen die Performance, zum andern eine technische Einschränkung in der Android Plattform. Die technische Einschränkung lässt sich folgendermaßen erklären: In den meisten Activites der Anwendung wird eine Instanz der Prozess Engine benötigt. Für die Realisierung wäre hierbei die *Referenzierung* auf genau *eine* Instanz der richtige Ansatz. Dazu müsste die Referenz beim Start einer Activity übergeben werden. Genau dieses ist jedoch nicht möglich, da als Übergabeparameter nur serialisierbare Objekte verwendet werden können.

Das bedeutet, dass alternativ an vielen Stellen Instanzen der Engine verwaltet werden müssen (erzeugen und verwerfen). Machbar wäre das, jedoch aber nicht effizient, da das Erzeugen einer Instanz der Prozess Engine gewisse Zeit in Anspruch nimmt. Die Lösung dafür ist der bereits beschriebene Einsatz eines Singleton Pattern. Dabei können beliebige Instanzen erzeugt werden, welche intern genau eine repräsentiert.

6.9.2. Nachrichtenorientierte Kommunikation

Ein weiterer interessanter Aspekt ist die *messageorientierte Kommunikation* mit der Prozess Engine. In den Schnittstellen sind Übergabeparameter und Rückgabewerte ausschließlich als Nachrichten definiert. Der Grund dafür ist, dass die Prozess Engine auf dem mobilen Endgerät beliebig austauschbar sein soll. Beispielsweise könnte man so auch direkt mit einer externen Prozess Engine kommunizieren, da diese die selbe Schnittstellendefinition aufweist.
Die einheitliche Übergabe der Werte als String bringt auch einige Nachteile:

- Die Typisierung geht verloren, da alle Objekt-Modelle serialisiert werden müssen. Dadurch ist nicht mehr implizit bekannt, ob ein Rückgabewert tatsächich einen String repräsentieren soll, oder eine Struktur.

- Durch die Aufhebung der Typisierung ist zusätzlicher Dokumentationsaufwand nötig.

Um diesen beschriebenen Problemen entgegen zu wirken, erhält jede Methode der Prozess Engine eine detaillierte Beschreibung der Übergabeparameter und der Rückgabewerte. Des Weiteren wird textuell beschrieben, mit welcher Methode eine Nachricht in ein Objekt konvertiert werden kann, und vor allem welchem Objekt-Typ die Nachricht entspricht.

6.9.3. Zustände der Knoten

Aus Sicht der mobilen Anwendung können die Knoten eines Templates die in Abbildung 6.5 dargestellten Zustände annehmen. Die in der Abbildung verwendeten Pfeiltexte stellen direkt die Schnittstellenfunktionen zur Prozess Engine dar.

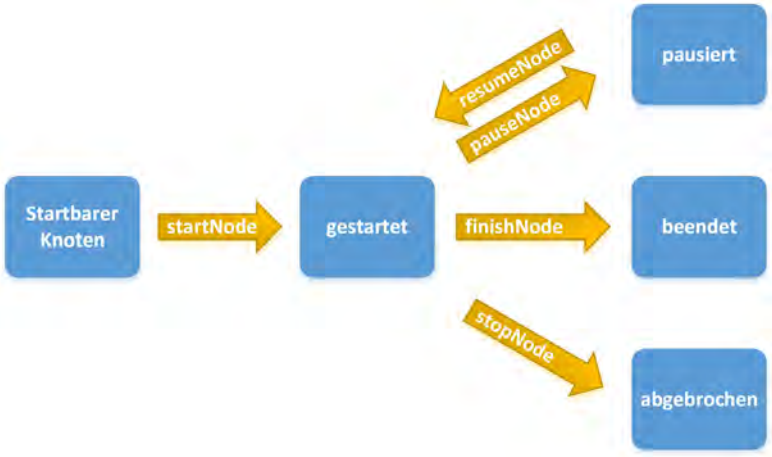


Abbildung 6.5.: Zustände eines Knotens aus Sicht der mobilen Anwendung

6.10. Probleme und Erkenntnisse bei der Implementierung

Wärend der Implementierungsphase sind einige Probleme aufgetreten, jedoch konnten alle gelöst werden. Im diesem Kapitel werden Problemstellungen beschrieben, deren Lösung längere Zeit in Anspruch nahm.

Nicht scrollbare Liste: Für die Realisierung der Startseite wird eine Liste benötigt, welche dynamisch Kindelemente aufnehmen kann, aber nicht scrollbar sein darf. Diese Liste muss immer in voller Höhe dargestellt werden, da es einen globalen `ScrollContainer` für alle Elemente auf dieser Seite gibt. Da diese Anforderung mit den Standard Elementen des Android Frameworks nicht umsetzbar ist, wurde für diesen Zweck ein eigenes Oberflächenelement realisiert (`ExpandableListItem`).

Benutzerdefinierte Titelzeile: In der hier dargelegten Arbeit werden Titelzeilen benötigt, welche nicht dem Android Style entsprechen. Das Android Framework bietet allerdings die Möglichkeit die Titlezeile durch ein eigens erstelltes Element zu ersetzen. Die Verwendung mehrerer verschiedener wird allerdings nicht unterstützt. Aus diesem Grund wird die Standard Android Titelzeile verwendet und in jeder

Aktivity die benötigten Elemente dynamisch hinzugefügt, oder bereits existerende ausgeblendet.

Animationen: Bei der Verwendung von Animationen unmittelbar nachdem die Benutzeroberfläche sichtbar wird kommt es zu Problemen in der Performance. Bilder werden dadurch nicht flüssig animiert. Dieses Problem ist jedoch auf die aktuell eingesetzte *Dalvik VM* zurückzuführen.

Process Engine: Die Kommunikation mit der Prozess Engine läuft streng nachrichtenorientiert ab. Dadurch gehen an den Schnittstellen die Typinformationen der Objekte verloren, da jedes Objekt als String serialisiert wird. Dies erschwert die Verwendung der Schnittstellen in der Anwendung. Es könnte jedoch eine Adapterschicht zwischen Prozess Engine und der Anwendungslogik implementiert werden, welche die selben Funktionen bereit stellt, jedoch als Übergabe- / Rückgabeparameter konkrete Objekte bereit stellt.

7

Zusammenfassung und Ausblick

Dieses Kapitel fasst die Arbeit zusammen und stellt dabei Neuerungen im Vergleich zu den bisherigen Implementierungen in den Vordergrund. Abschließend wird ein Ausblick über Verbesserungen oder mögliche Erweiterungen gegeben.

7.1. Zusammenfassung

Im Rahmen der hier vorgelegten Arbeit wurde ein Konzept für eine dynamisch generierte Anwendung für prozess-orientierte Fragebögen am Beispiel der Android Plattform entworfen und anschließend realisiert. Dies ist Teil eines Gesamt-Systems, welches sich in die drei Komponenten Konfigurator, Server und mobiler Client unterteilt. Mit dem Konfigurator können Fragebögen erstellt, mit dem Server gespeichert und mit der Client-Anwendung ausgeführt werden. Hierbei gab es einige Anforderungen an die

Anwendung, welche erfüllt werden mussten. Diese Anforderungen deckten sich Teils mit den Anforderungen des Gesamt-Systems und sind zum anderen Teil spezifisch auf den Fragebogen Client zu beziehen.

Eine der wichtigsten Anforderungen an die zu realisierende Anwendung war die dynamische Generierung einer Benutzeroberfläche anhand einer Seitenbeschreibung. Die Seitenbeschreibung ist in einem Fragebogen enthalten und wird auf einer Konfigurationsumgebung modelliert. Dieses Vorgehen hat den Vorteil, dass zur Darstellung neuer oder geänderter Fragebögen, die Anwendung auf dem mobilen Endgerät nicht erneuert werden muss.
Ein weiterer Vorteil ist die Anforderung der Prozessunterstützung für den Fragebogenablauf. Dazu wird eine Prozess Engine eingesetzt, welche sich lokal auf dem mobilen Endgerät befindet, um autonom Umfragen durchzuführen. Die Engine kann komplexe Ablauflogik interpretieren und dadurch den Fragebogenablauf steuern. Die dazu vorausgesetzte Ablauflogik ist ebenfalls Teil des Fragebogens und deshalb nicht im Programmcode enthalten.
Des Weiteren wurde in der Architektur vorgesehen, Fragebögen in einer Datenbank auf dem mobilen Endgerät zu persistieren. Durch diese Maßnahme werden Offline-Umfragen unterstützt, da alle für die Umfrage notwendigen Ressourcen auf dem mobilen Endgerät vorhanden sind (Fragebogen und Prozess Engine). Somit sind Umfragen an verschiedensten Orten möglich, da für die Durchführung keine Internetverbindung vorausgesetzt wird. Die Durchführung von Umfragen in unterschiedlichen Bereichen war ebenfalls eine wichtige Anforderung. Dadurch ist es möglich, die Anwendung flächendeckend einzusetzen und nicht nur bereichsspezifisch, wie beispielsweise im Bereich Medizin. Um dieser Anforderung nachzukommen, wurde die Benutzeroberfläche in sehr neutralem Design gehalten. Zusätzlich wurde für jedes eingesetzte Interaktionselement ein initialer neutraler Zustand eingeführt, um den Benutzer im Entscheidungsprozess durch bereits ausgewählte Antworten nicht zu beeinflussen.
Die realisierte Anwendung unterstützt derzeit nur die Interaktionselemente, welche im Kapitel 4.6 definiert wurden. Durch den modularen Aufbau der Anwendung können jedoch leicht weitere Interaktionselemente für neue Fragen-Typen integriert werden.
Umfragen werden häufig in Ländern unterschiedlicher Sprachen durchgeführt. Somit gilt

die Sprachumschaltung als Grundanforderung. In der realisierten Anwendung können die Sprache der Anwendung und des Fragebogens getrennt voneinander geändert werden. Die ist ein großer Vorteil, da Probanden die Fragebogen in der jeweiligen Landessprache bearbeiten können.

Zusammenfassend kann festgehalten werden, dass die Realisierung der Anwendung unter Berücksichtigung der in Kapitel 2 diskutierten Nachteile sehr gelungen ist. Trotzdem ist noch sehr viel Potential zur Weiterentwicklung vorhanden. Einige Beispiele dafür werden in Kapitel 7.2 aufgezeigt.

7.2. Ausblick

Das Kapitel Ausblick enthält Anregungen für künftige Weiterentwicklungen, sowie Beispiele von alternativen Anwendungsmöglichkeiten des Fragebogensystems. Nach dem eine Umfrage abgeschlossen ist, wäre es hilfreich die Reihenfolge der durchgeführten Interviews nachträglich zu betrachten. Dazu könnte der administrative Bereich um diese Möglichkeit erweitert werden. Dabei könnte auch die Reihenfolge der beantworteten Fragen dargestellt werden.

Des Weiteren ist in der Architektur der Anwendung die Verwendung von Sensoren berücksichtigt, um die vom Benutzer erhobenen Antworten um zusätzliche Informationen zu erweitern. Üblicherweise sind das Mikrofon und Kamera, welche fest im mobilen Endgerät verbaut sind. Dadurch wird die Aussagekraft der Antworten vom Benutzer verstärkt. Jedoch ist dem Konfigurator bei der Erstellung eines Fragbogens nicht bekannt welche Sensoren ein mobiles Endgerät besitzt. Dazu wäre im administrativen Bereich eine extra Seite denkbar, auf welcher die verfügbaren Sensoren verwaltet werden. Beispielsweise könnten Typen von Sensoren definiert werden, welchen an dieser Stelle reale Sensoren zugewiesen werden. Ebenso wäre denkbar, externe Sensoren zu verwenden, welche in der Regel kabellos mit dem mobile Endgerät verbunden werden. So könnten zusätzliche Vitalparameter (beispielsweise Blutdruck und Sauerstoffsättigung) zu einer Antwort gespeichert werden. Zusätzlich ist hier zwischen zwei Arten der Erfassungsfreqenz zu unterscheiden. Diese kann einmalig (fragenabhängige Bildaufnahme), oder kontinuier-

lich (seitenabhängige Audioaufzeichnung) erfolgen. Somit ergeben sich unterschiedliche Anforderungen an die Speicherung dieser Daten. Zum einen beispielsweise als Einzelwert und zum anderen als Bereich diskreter Einzelwerte. Realisierbar ist die Anbindung von Sensoren beispielsweise durch Verwendung des im Rahmen einer Abschlussarbeit entstandenen Sensorframeworks [Zel13]. Dieses vereinheitlicht auf Basis der Android Plattform den Zugriff auf Sensoren. Dabei ist nicht mehr auf verschieden Protokolle, oder Kommunikationstypen (beispielsweise Bluetooth oder USB) zu achten.

Das Fragebogensystem ist darauf ausgelegt, in unterschiedlichen Domänen eingesetzt zu werden. Dabei ist denkbar, dass Anforderungen für neue Interaktionselemente formuliert werden. Dies könnte die Anzeige von Bilder, oder ein neuer Typ für Fragen sein. Die realisierte Anwendung ist bereits bei der Architektur so modular konzipiert worden, dass das Einfügen neuer Interaktionselemente sehr gut unterstützt wird. Realisiert wird ein neues Element für die Oberfläche, indem durch Ableiten von `QuestionListItem` eine neue Klasse erstellt wird. Diese implementiert das neue Interaktionselement. Anschließend wird beim User Interface Generator der neue Typ eingeführt. Mit diesem Vorgehen können beliebige neue Elemente integriert werden.

In dem hier dargelegten Kontext wird das System ausschließlich zur Durchführung von Umfragen verwendet. Bei genauerem Betrachten unterstützt das System jedoch die Steuerung durch einen modellierten Prozess, die Protokollierung der eingegebenen Daten und eine Validierung auf den Erhalt von Antworten. Unter diesen Voraussetzungen kann das System ebenso für prozessgesteuerte Checklisten verwendet werden, welche beispielsweise bei Reparaturen eingesetzt werden. Hierbei sind die einzelnen durchzuführenden Schritte in zeitlicher Reihenfolge zu modellieren. Dabei entspricht ein Schritt oder eine Schrittkette genau einer Seite. Auf die nächste Seite kann erst gewechselt werden, wenn der aktuelle Schritt durchgeführt ist (dies könnte mit einem Zweistellungsschalter umgesetzt werden). Eine andere denkbare Domäne wäre der medizinische Bereich. Bei der Aufnahme von Patienten könnten so die einzelnen Aufnahmeschritte als Prozess definiert werden. Dem Patienten wird dabei das mobile Endgerät zur Verfügung gestellt, wodurch er die einzelnen zu besuchenden Stationen in chronologischer Reihenfolge angezeigt bekommt.

Abschließend kann zusammengefasst werden, dass die realisierte Anwendung sehr leicht erweiterbar ist und in den verschiedensten Domänen ihren Einsatz finden kann. Die größte Hürde, die es dabei zu Überwinden gilt, ist sicherlich die Akzeptanz der Anwender.

A

Styleguide

In diesem Anhang ist der Styleguide aufgeführt, welcher zur Realisierung verwendet wurde [Sch14].

Questionsys

MiniStyleguide
Gesamtsystem

Einleitung

Das Projekt

Die Universität Ulm und die Universität Konstanz realisieren gemeinsam ein Projekt, das sich mit der mobilen und flexiblen Datenerhebung anhand generischer Fragebögen befasst.

Die Subsysteme

Das Projekt besteht aus drei Modulen, die jeweils selbstständig agieren jedoch von einander abhängen. Um die logische Zusammengehörigkeit zu visualisieren, wurden die einzelnen Logos grafisch auf eine gemeinsame Basis gestellt.

Corporate Design

Durch ein einheitliches Erscheinungsbild soll der Wiedererkennungswert gesteigert, sowie eine einheitliche Bedienung auf den verschiedensten Geräten garantiert werden.

Autor
Steffen Scherle
steffen@scherle.de

Betreuer
Johannes Schobel
johannes.schobel@uni-ulm.de
Marc Schickler
marc.schickler@uni-ulm.de
Rüdiger Pryss
ruediger.pryss@uni-ulm.de

Universität Ulm
Fakultät für Ingenieurwissenschaften und Informatik
Institut für Datenbanken und Informationssysteme

Inhalt

I.

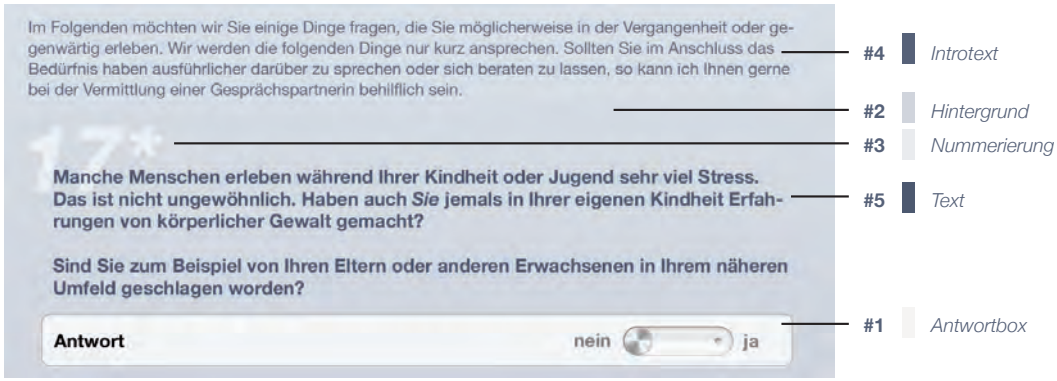
Inhalt

Farben

	R	G	B	Web	C	M	Y	K	Einsatz
#1	247	247	247	#F7F7F7	3	3	3	0	*Antwortbox*
#2	238	239	241	#EEEFF1	8	5	4	0	*Nummerierung*
#3	218	220	226	#DADCE2	17	12	8	0	*Hintergrund*
#4	87	96	117	#576075	70	58	36	15	*Introtext*
#5	76	86	108	#4C566C	74	61	38	20	*Text*

Konzept

Da das System unter anderem auch für Befragung psychisch erkrankter Personen verwendet wird, wurde absichtlich auf den Einsatz auffälliger und emotionsgebundener Farben (wie z.B. Rot) verzichtet. Daher muss bei der Implementierung darauf geachtet werden die hier aufgeführten Farben zu verwenden.



Beispiel des Einsatzes von Farben

Die Datenerfassung erfolgt auf mobilen Endgeräten (z.B. Tablets oder Smartphones) mittels Touch-Eingabe. Das Beispiel zeigt die Darstellung einer Frage mit zugehöriger Antwortoption. Hier ist der Einsatz aller fünf Basisfarben erkennbar.

II.

Inhalt

Typografie

	Schriftart	Größe	Farbe	Einsatz
ABCDEabcde123	Helvetica Neue Bold	34 Punkt	#5 *(#4C566C)*	*Text*
ABCDEabcde123	Helvetica Neue Bold	34 Punkt	schwarz *(#000000)*	*Antwort*
123	Helvetica Neue Bold Italic	170 Punkt	#2 *(#EEEFF1)*	*Nummerierung groß*
123	Helvetica Neue Bold Italic	83 Punkt	#2 *(#EEEFF1)*	*Nummerierung klein*
ABCDEabcde123	Helvetica Neue Regular	30 Punkt	#4 *(#576075)*	*Intro*

Effekt: Schlagschatten / #FFFFFF / 75% Opacity / 90° Ausrichtung / Abstand 10%

III.

Inhalt

Farbe #5
RGB: 76 86 108
Web: #4C566C
CMYK: 74 61 38 20

Negativbeispiel:
verschiedene i-Punkte
auf verschiedener Höhe

Questionnaire

runde Form:
Gill Sans Bold

„i-Punkt“:
entfernt

„uestion“:
Gill Sans Bold (220pt)

Vektorgrafik:
„?“ trifft auf „Q“

Postfix:
Gill Sans Light (244pt)

Subbranding

Logo Client

Logo Konfigurator

Logo Verwaltung

Einsatz ohne Untertitel:
intern (innerhalb des jeweiligen Systems)

Einsatz mit Untertitel:
extern (bei gleichzeitiger Nennung mehrerer Systeme)

Subbranding

Questionnaire der Fragebogen-Client — aus dem Englischen „Questionnaire“ für deutsch „Fragebogen“

Questioneer der Konfigurator — Kunstwort aus „Questionnaire“ und „Engineer“

Questionizer Die Verwaltung — Kunstwort aus „Questionnaire“ und „Organizer“

Q-Form
Schlagschatten: schwarz | 62% Sichtbarkeit
2px Abstand | 90°
Innenschatten: weiß | 75% Sichtbarkeit
2px Abstand | 90°
Verlaufsfüllung: #000 - #FFF | 21% Deckkraft
linear | 90° | 57% skaliert

App Icon

Hintergrundfarbe
RGB: 15 129 198
Web: #0f81c6
+ Apple‘s Presupplied iOS-Gloss

„i-Punkt“:
entfernt

Untertitel:
Myriad Pro (47pt)
50% Opacity

Untertitel:
Blockausrichtung
mit Postfix-Teil

Questionsys

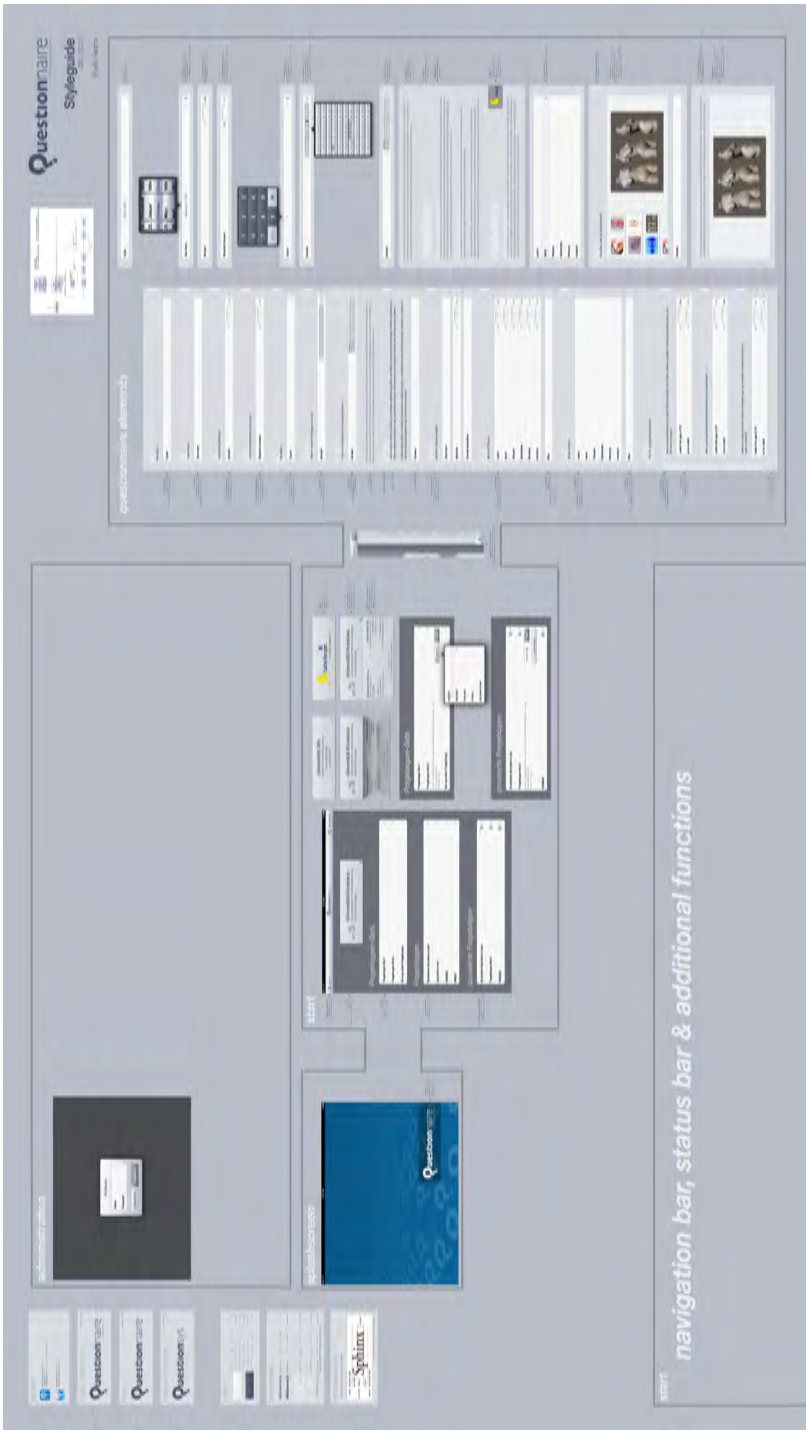


Abbildung A.1.: User Interface Entwurf

B

Schnittstellen der Prozess Engine

In diesem Anhang sind die Schnittstellen zur Prozess Engine aufgeführt.

- storeTemplate(String template)
- deleteTemplate(string templateID)
- startInstance(String templateID)
- stopInstance(String instanceID)
- pauseInstance(String instanceID)
- resumeInstance(String instanceID)
- deleteInstance(String instanceID)
- getTemplates()
- getInstances()

- getInstances(String templateID)
- getInstances(String status)
- getInstances(String templateID, String status)
- getStartableNode(String instanceID)
- startNode(String instanceID, String nodeID)
- stopNode(String instanceID, String nodeID)
- pauseNode(String instanceID, String nodeID)
- resumeNode(String instanceID, String nodeID)
- finishNode(String instanceID, String nodeID)
- setValueForDataElement(String instanceID, String nodeID, String dataElementID, String value)
- setValueForDataElementByName (String instanceID, String nodeID, String dataElementName, String value)
- getHistory(String instanceID) getData(String instanceID)

C

Seitenbeschreibung

In diesem Anhang ist ein beispielhafter Inhalt einer Seitenbeschreibung in JSON gelistet, welcher Beschreibungen für alle unterstützten Typen von Fragen aufzeigt. Dieser wird als Basis zur dynamischen Generierung der Benutzeroberfläche durch den User Interface Generator verwendet.

```
"page": {
  "idHigh": "12345",
  "idLow": "12345",
  "version": "1",
  "metadata": "die MetaData !",
  "elements": [
    {
      "idHigh": "9991",
      "idLow": "9991",
      "version": "5",
      "dateCreated": "1386143937",
      "dateLastModified": "1386143937",
```

```
13        "name": "t_rauchen",
14        "friendlyName": "Titel Rauchen",
15        "type" : "HEADLINE",
16        "headline": "{
17          \"languageTextContainer\": [
18            {
19              \"languageCode\": \"de-de\",
20              \"languageValue\": \"Rauchen\"
21            },
22            {
23              \"languageCode\": \"en-us\",
24              \"languageValue\": \"Smoking\"
25            }
26          ]
27        }"
28      },
29      {
30        "idHigh": "1000",
31        "idLow": "1000",
32        "version": "2",
33        "dateCreated": "1386143937",
34        "dateLastModified": "1386143937",
35        "name": "p_rauchen",
36        "friendlyName": "Text Rauchen",
37        "type" : "TEXT",
38        "headline": "{
39          \"languageTextContainer\": [
40            {
41              \"languageCode\": \"de-de\",
42              \"languageValue\": \"Rauchen\"
43            },
44            {
45              \"languageCode\": \"en-us\",
46              \"languageValue\": \"Smoking\"
47            }
48          ]
49        }",
50        "text": "{
51          \"languageTextContainer\": [
52            {
53              \"languageCode\": \"de-de\",
54              \"languageValue\": \"TEXT.\"
55            },
56            {
57              \"languageCode\": \"en-us\",
```

```
            \"languageValue\": \"This is an introductory text about smoking
                ...\"
          }
        ]
      }"
    },
    {
      "idHigh": "1001",
      "idLow": "1001",
      "version": "2",
      "dateCreated": "1386143937",
      "dateLastModified": "1386143937",
      "name": "p_rauchen",
      "friendlyName": "Text Rauchen",
      "type" : "TEXT",
      "headline": "{
        \"languageTextContainer\": [
          {
            \"languageCode\": \"de-de\",
            \"languageValue\": \"Paragraph 2\"
          },
          {
            \"languageCode\": \"en-us\",
            \"languageValue\": \"Paragraph 2\"
          }
        ]
      }",
      "text": "{
        \"languageTextContainer\": [
          {
            \"languageCode\": \"de-de\",
            \"languageValue\": \"TEXT\"
          },
          {
            \"languageCode\": \"en-us\",
            \"languageValue\": \"TEXT\"
          }
        ]
      }"
    },
    {
      "idHigh": "1002",
      "idLow": "1002",
      "version": "2",
      "dateCreated": "1386143937",
```

```
      "dateLastModified": "1386143937",
      "name": "p_rauchen",
      "friendlyName": "Text Rauchen",
      "type" : "TEXT",
      "text": "{
        \"languageTextContainer\": [
          {
            \"languageCode\": \"de-de\",
            \"languageValue\": \"TEXT\"
          },
          {
            \"languageCode\": \"en-us\",
            \"languageValue\": \"TEXT\"
          }
        ]
      }"
    },
    {
      "idHigh": "1003",
      "idLow": "1003",
      "version": "1",
      "dateCreated": "1386143937",
      "dateLastModified": "1386143937",
      "name": "q_rauchen",
      "friendlyName": "Rauchen Sie?",
      "type" : "QUESTION",
      "text": "{
        \"languageTextContainer\": [
          {
            \"languageCode\": \"de-de\",
            \"languageValue\": \"Nicht vorausgew hlter Zweistellungsschalter
                 (BOOLEAN)\"
          },
          {
            \"languageCode\": \"en-us\",
            \"languageValue\": \"Unselected Boolean (BOOLEAN)\"
          }
        ]
      }",
      "questions": "[
        {
          \"questionType\": \"BOOLEAN\",
          \"required\": \"true\",
          \"dataElement\": \"de_3\",
          \"text\": {
```

```
            \"languageTextContainer\": [
              {
                \"languageCode\": \"de-de\",
                \"languageValue\": \"Antwort Text\"
              },
              {
                \"languageCode\": \"en-us\",
                \"languageValue\": \"Answer Text\"
              }
            ]
          },
          \"itemList\" : [
            {
              \"dataValue\": \"0\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"nein\"
                },
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"no\"
                }
              ]
            },
            {
              \"dataValue\": \"1\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"ja\"
                },
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"yes\"
                }
              ]
            }
          ]
        }
      ]"
    },
    {
      "idHigh": "1004",
      "idLow": "1004",
```

```
191       "version": "1",
192       "dateCreated": "1386143937",
193       "dateLastModified": "1386143937",
194       "name": "q_rauchen",
195       "friendlyName": "Rauchen Sie?",
196       "type" : "QUESTION",
197       "text": "{
198         \"languageTextContainer\": [
199           {
200             \"languageCode\": \"de-de\",
201             \"languageValue\": \"Nicht vorausgew hlter Zweistellungsschalter
                    mit neutralem Bereich (BOOLEAN_EXTENDED)\"
202           },
203           {
204             \"languageCode\": \"en-us\",
205             \"languageValue\": \"Unselected Boolean with neutral state (
                   BOOLEAN_EXTENDED)\"
206           }
207         ]
208       }",
209       "questions": "[
210         {
211           \"questionType\": \"BOOLEAN_EXTENDED\",
212           \"required\": \"false\",
213           \"dataElement\": \"de_4\",
214           \"text\": {
215             \"languageTextContainer\": [
216               {
217                 \"languageCode\": \"de-de\",
218                 \"languageValue\": \"Antwort Text\"
219               },
220               {
221                 \"languageCode\": \"en-us\",
222                 \"languageValue\": \"Answer Text\"
223               }
224             ]
225           },
226           \"itemList\" : [
227             {
228               \"dataValue\": \"0\",
229               \"languageTextContainer\": [
230                 {
231                   \"languageCode\": \"de-de\",
232                   \"languageValue\": \"nein\"
233                 },
```

```
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"no\"
                }
              ]
            },
            {
              \"dataValue\": \"1\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"ja\"
                },
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"yes\"
                }
              ]
            }
          ]
        }
      ]"
    },
    {
      "idHigh": "1005",
      "idLow": "1005",
      "version": "7",
      "dateCreated": "1386143937",
      "dateLastModified": "1386143937",
      "name": "q_rauchen_anzahl",
      "friendlyName": "Anzahl Zigaretten",
      "type" : "QUESTION",
      "text": "{
        \"languageTextContainer\": [
          {
            \"languageCode\": \"de-de\",
            \"languageValue\": \"Das ist ein Slider Beispiel (SLIDER)\"
          },
          {
            \"languageCode\": \"en-us\",
            \"languageValue\": \"This is an example of the use of sliders (
                SLIDER)\"
          }
        ]
      }",
```

```
278       "questions": "[
279         {
280           \"questionType\": \"SLIDER\",
281           \"required\": \"true\",
282           \"dataElement\": \"de_5\",
283           \"text\": {
284             \"languageTextContainer\": [
285               {
286                 \"languageCode\": \"de-de\",
287                 \"languageValue\": \"Antwort Text\"
288               },
289               {
290                 \"languageCode\": \"en-us\",
291                 \"languageValue\": \"Answer Text\"
292               }
293             ]
294           },
295           \"itemList\" : [
296             {
297               \"dataValue\": \"1\",
298               \"languageTextContainer\": [
299                 {
300                   \"languageCode\": \"de-de\",
301                   \"languageValue\": \"1\"
302                 },
303                 {
304                   \"languageCode\": \"en-us\",
305                   \"languageValue\": \"1\"
306                 }
307               ]
308             },
309             {
310               \"dataValue\": \"2\",
311               \"languageTextContainer\": [
312                 {
313                   \"languageCode\": \"de-de\",
314                   \"languageValue\": \"2\"
315                 },
316                 {
317                   \"languageCode\": \"en-us\",
318                   \"languageValue\": \"2\"
319                 }
320               ]
321             },
322             {
```

```
              \"dataValue\": \"3\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"3\"
                },
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"3\"
                }
              ]
            },
            {
              \"dataValue\": \"4\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"4\"
                },
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"4\"
                }
              ]
            },
            {
              \"dataValue\": \"5\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"5\"
                },
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"5\"
                }
              ]
            }
          ]
        }
      ]"
    },
    {
      "idHigh": "1006",
      "idLow": "1006",
```

```
368       "version": "3",
369       "dateCreated": "1386143937",
370       "dateLastModified": "1386143937",
371       "name": "q_rauchen_marke",
372       "friendlyName": "Zigarettenmarke?",
373       "type" : "QUESTION",
374       "text": "{
375         \"languageTextContainer\": [
376           {
377             \"languageCode\": \"de-de\",
378             \"languageValue\": \"Hier kann ein Text eingegeben werden (
                    TEXT_INPUT)\"
379           },
380           {
381             \"languageCode\": \"en-us\",
382             \"languageValue\": \"A text can be entered here (TEXT_INPUT)\"
383           }
384         ]
385       }",
386       "questions": "[
387         {
388           \"questionType\": \"TEXT_INPUT\",
389           \"required\": \"false\",
390           \"dataElement\": \"de_6\",
391           \"text\": {
392             \"languageTextContainer\": [
393               {
394                 \"languageCode\": \"de-de\",
395                 \"languageValue\": \"Antwort Text\"
396               },
397               {
398                 \"languageCode\": \"en-us\",
399                 \"languageValue\": \"Answer Text\"
400               }
401             ]
402           }
403         }
404       ]"
405     },
406     {
407       "idHigh": "1007",
408       "idLow": "1007",
409       "version": "3",
410       "dateCreated": "1386143937",
411       "dateLastModified": "1386143937",
```

```
412       "name": "q_rauchen_marke",
413       "friendlyName": "Zigarettenmarke?",
414       "type" : "QUESTION",
415       "text": "{
416         \"languageTextContainer\": [
417           {
418             \"languageCode\": \"de-de\",
419             \"languageValue\": \"Hier kann eine Zahl eingegeben werden (
                  NUMBER_INPUT)\"
420           },
421           {
422             \"languageCode\": \"en-us\",
423             \"languageValue\": \"A number can be entered here (NUMBER_INPUT)
                  \"
424           }
425         ]
426       }",
427       "questions": "[
428         {
429           \"questionType\": \"NUMBER_INPUT\",
430           \"required\": \"true\",
431           \"dataElement\": \"de_7\",
432           \"text\": {
433             \"languageTextContainer\": [
434               {
435                 \"languageCode\": \"de-de\",
436                 \"languageValue\": \"Antwort Text\"
437               },
438               {
439                 \"languageCode\": \"en-us\",
440                 \"languageValue\": \"Answer Text\"
441               }
442             ]
443           }
444         }
445       ]"
446     },
447     {
448       "idHigh": "1008",
449       "idLow": "1008",
450       "version": "3",
451       "dateCreated": "1386143937",
452       "dateLastModified": "1386143937",
453       "name": "q_rauchen_marke",
454       "friendlyName": "Zigarettenmarke?",
```

```
455       "type" : "QUESTION",
456       "text": "{
457         \"languageTextContainer\": [
458           {
459             \"languageCode\": \"de-de\",
460             \"languageValue\": \"Hier kann ein Datum eingegeben werden (
                    DATE_INPUT)\"
461           },
462           {
463             \"languageCode\": \"en-us\",
464             \"languageValue\": \"A date can be entered here (DATE_INPUT)\"
465           }
466         ]
467       }",
468       "questions": "[
469         {
470           \"questionType\": \"DATE_INPUT\",
471           \"required\": \"true\",
472           \"dataElement\": \"de_8\",
473           \"text\": {
474             \"languageTextContainer\": [
475               {
476                 \"languageCode\": \"de-de\",
477                 \"languageValue\": \"Antwort Text\"
478               },
479               {
480                 \"languageCode\": \"en-us\",
481                 \"languageValue\": \"Answer Text\"
482               }
483             ]
484           }
485         }
486       ]"
487     },
488     {
489       "idHigh": "1009",
490       "idLow": "1009",
491       "version": "3",
492       "dateCreated": "1386143937",
493       "dateLastModified": "1386143937",
494       "name": "q_rauchen_marke",
495       "friendlyName": "Zigarettenmarke?",
496       "type" : "QUESTION",
497       "text": "{
498         \"languageTextContainer\": [
```

```
499           {
500             \"languageCode\": \"de-de\",
501             \"languageValue\": \"W hlen sie die Bereiche aus (RANGE_INPUT)\"
502           },
503           {
504             \"languageCode\": \"en-us\",
505             \"languageValue\": \"Please select the ranges you like (
                    RANGE_INPUT)\"
506           }
507         ]
508       }",
509       "questions": "[
510         {
511           \"questionType\": \"RANGE_INPUT\",
512           \"required\": \"true\",
513           \"dataElement\": \"de_9\",
514           \"text\": {
515             \"languageTextContainer\": [
516               {
517                 \"languageCode\": \"de-de\",
518                 \"languageValue\": \"Antwort Text\"
519               },
520               {
521                 \"languageCode\": \"en-us\",
522                 \"languageValue\": \"Answer Text\"
523               }
524             ]
525           },
526           \"minValue\": \"1\",
527           \"maxValue\": \"30\"
528         }
529       ]"
530     },
531     {
532       "idHigh": "1009",
533       "idLow": "1009",
534       "version": "3",
535       "dateCreated": "1386143937",
536       "dateLastModified": "1386143937",
537       "name": "q_rauchen_marke",
538       "friendlyName": "Zigarettenmarke?",
539       "type" : "QUESTION",
540       "text": "{
541         \"languageTextContainer\": [
542           {
```

```
543             \"languageCode\": \"de-de\",
544             \"languageValue\": \"W h l e n  sie die Bereiche aus (RANGE_INPUT)\"
545           },
546           {
547             \"languageCode\": \"en-us\",
548             \"languageValue\": \"Please select the ranges you like (
                    RANGE_INPUT)\"
549           }
550         ]
551       }",
552       "questions": "[
553         {
554           \"questionType\": \"RANGE_INPUT\",
555           \"required\": \"true\",
556           \"dataElement\": \"de_9_1\",
557           \"text\": {
558             \"languageTextContainer\": [
559               {
560                 \"languageCode\": \"de-de\",
561                 \"languageValue\": \"Antwort Text\"
562               },
563               {
564                 \"languageCode\": \"en-us\",
565                 \"languageValue\": \"Answer Text\"
566               }
567             ]
568           },
569           \"minValue\": \"1\",
570           \"maxValue\": \"30\"
571         }
572       ]"
573     },
574     {
575       "idHigh": "1010",
576       "idLow": "1010",
577       "version": "2",
578       "dateCreated": "1386143937",
579       "dateLastModified": "1386143937",
580       "name": "p_rauchen",
581       "friendlyName": "Text Rauchen",
582       "type" : "TEXT",
583       "text": "{
584         \"languageTextContainer\": [
585           {
586             \"languageCode\": \"de-de\",
```

```
            \"languageValue\": \"TEXT\"
          },
          {
            \"languageCode\": \"en-us\",
            \"languageValue\": \"TEXT\"
          }
        ]
      }"
    },
    {
      "idHigh": "1011",
      "idLow": "1011",
      "version": "7",
      "dateCreated": "1386143937",
      "dateLastModified": "1386143937",
      "name": "q_rauchen_anzahl",
      "friendlyName": "Anzahl Zigaretten",
      "type" : "QUESTION",
      "text": "{
        \"languageTextContainer\": [
          {
            \"languageCode\": \"de-de\",
            \"languageValue\": \"Einzelauswahl als aufklappbare Box (
                SINGLE_CHOICE)\"
          },
          {
            \"languageCode\": \"en-us\",
            \"languageValue\": \"Single choice as drop down box (
                SINGLE_CHOICE)\"
          }
        ]
      }",
      "questions": "[
        {
          \"questionType\": \"SINGLE_CHOICE\",
          \"required\": \"true\",
          \"dataElement\": \"de_11\",
          \"text\": {
            \"languageTextContainer\": [
              {
                \"languageCode\": \"de-de\",
                \"languageValue\": \"Antwort Text\"
              },
              {
                \"languageCode\": \"en-us\",
```

```
630                 \"languageValue\": \"Answer Text\"
631               }
632             ]
633           },
634           \"itemList\" : [
635             {
636               \"dataValue\": \"1\",
637               \"languageTextContainer\": [
638                 {
639                   \"languageCode\": \"de-de\",
640                   \"languageValue\": \"Eins\"
641                 },
642                 {
643                   \"languageCode\": \"en-us\",
644                   \"languageValue\": \"One\"
645                 }
646               ]
647             },
648             {
649               \"dataValue\": \"2\",
650               \"languageTextContainer\": [
651                 {
652                   \"languageCode\": \"de-de\",
653                   \"languageValue\": \"Zwei\"
654                 },
655                 {
656                   \"languageCode\": \"en-us\",
657                   \"languageValue\": \"Two\"
658                 }
659               ]
660             },
661             {
662               \"dataValue\": \"3\",
663               \"languageTextContainer\": [
664                 {
665                   \"languageCode\": \"de-de\",
666                   \"languageValue\": \"Drei\"
667                 },
668                 {
669                   \"languageCode\": \"en-us\",
670                   \"languageValue\": \"Three\"
671                 }
672               ]
673             },
674             {
```

```
675               \"dataValue\": \"4\",
676               \"languageTextContainer\": [
677                 {
678                   \"languageCode\": \"de-de\",
679                   \"languageValue\": \"Vier\"
680                 },
681                 {
682                   \"languageCode\": \"en-us\",
683                   \"languageValue\": \"Four\"
684                 }
685               ]
686             }
687           ]
688         }
689       ]"
690     },
691     {
692       "idHigh": "1012",
693       "idLow": "1012",
694       "version": "3",
695       "dateCreated": "1386143937",
696       "dateLastModified": "1386143937",
697       "name": "q_rauchen_marke",
698       "friendlyName": "Zigarettenmarke?",
699       "type" : "QUESTION_BLOCK",
700       "text": "{
701         \"languageTextContainer\": [
702           {
703             \"languageCode\": \"de-de\",
704             \"languageValue\": \"Block aus verschiedenen Fragen (
                    QUESTION_BLOCK)\"
705           },
706           {
707             \"languageCode\": \"en-us\",
708             \"languageValue\": \"Multiple answer block (QUESTION_BLOCK)\"
709           }
710         ]
711       }",
712       "questions": "[
713         {
714           \"questionType\": \"DATE_INPUT\",
715           \"required\": \"true\",
716           \"dataElement\": \"de_12_1\",
717           \"text\": {
718             \"languageTextContainer\": [
```

```
719               {
720                 \"languageCode\": \"de-de\",
721                 \"languageValue\": \"Geburtsjahr\"
722               },
723               {
724                 \"languageCode\": \"en-us\",
725                 \"languageValue\": \"Year born\"
726               }
727             ]
728           }
729         },
730         {
731           \"questionType\": \"BOOLEAN\",
732           \"required\": \"true\",
733           \"dataElement\": \"de_12_2\",
734           \"text\": {
735             \"languageTextContainer\": [
736               {
737                 \"languageCode\": \"de-de\",
738                 \"languageValue\": \"Analphabet\"
739               },
740               {
741                 \"languageCode\": \"en-us\",
742                 \"languageValue\": \"Illiterate\"
743               }
744             ]
745           },
746           \"itemList\" : [
747             {
748               \"dataValue\": \"0\",
749               \"languageTextContainer\": [
750                 {
751                   \"languageCode\": \"de-de\",
752                   \"languageValue\": \"nein\"
753                 },
754                 {
755                   \"languageCode\": \"en-us\",
756                   \"languageValue\": \"no\"
757                 }
758               ]
759             },
760             {
761               \"dataValue\": \"1\",
762               \"languageTextContainer\": [
763                 {
```

```
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"ja\"
                },
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"yes\"
                }
              ]
            }
          ]
        },
        {
          \"questionType\": \"TEXT_INPUT\",
          \"required\": \"false\",
          \"dataElement\": \"de_12_3\",
          \"text\": {
            \"languageTextContainer\": [
              {
                \"languageCode\": \"de-de\",
                \"languageValue\": \"Ausbildung\"
              },
              {
                \"languageCode\": \"en-us\",
                \"languageValue\": \"Education\"
              }
            ]
          }
        }
      ]"
    },
    {
      "idHigh": "1013",
      "idLow": "1013",
      "version": "3",
      "dateCreated": "1386143937",
      "dateLastModified": "1386143937",
      "name": "q_rauchen_marke",
      "friendlyName": "Zigarettenmarke?",
      "type" : "QUESTION_GROUP",
      "text": "{
        \"languageTextContainer\": [
          {
            \"languageCode\": \"de-de\",
            \"languageValue\": \"Gruppe aus selektiv Fragen (QUESTION_GROUP)
                \"
```

```
808           },
809           {
810             \"languageCode\": \"en-us\",
811             \"languageValue\": \"Multiple choice group (QUESTION_GROUP)\"
812           }
813         ]
814       }",
815       "questions": "[
816         {
817           \"questionType\": \"BOOLEAN\",
818           \"required\": \"true\",
819           \"dataElement\": \"de_13_1\",
820           \"text\": {
821             \"languageTextContainer\": [
822               {
823                 \"languageCode\": \"de-de\",
824                 \"languageValue\": \"Justo\"
825               },
826               {
827                 \"languageCode\": \"en-us\",
828                 \"languageValue\": \"Justo\"
829               }
830             ]
831           },
832           \"itemList\" : [
833             {
834               \"dataValue\": \"0\",
835               \"languageTextContainer\": [
836                 {
837                   \"languageCode\": \"de-de\",
838                   \"languageValue\": \"nein\"
839                 },
840                 {
841                   \"languageCode\": \"en-us\",
842                   \"languageValue\": \"no\"
843                 }
844               ]
845             },
846             {
847               \"dataValue\": \"1\",
848               \"languageTextContainer\": [
849                 {
850                   \"languageCode\": \"de-de\",
851                   \"languageValue\": \"ja\"
852                 },
```

```
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"yes\"
                }
              ]
            }
          ]
        },
        {
          \"questionType\": \"BOOLEAN\",
          \"required\": \"true\",
          \"dataElement\": \"de_13_2\",
          \"text\": {
            \"languageTextContainer\": [
              {
                \"languageCode\": \"de-de\",
                \"languageValue\": \"Amet\"
              },
              {
                \"languageCode\": \"en-us\",
                \"languageValue\": \"Amet\"
              }
            ]
          },
          \"itemList\" : [
            {
              \"dataValue\": \"0\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"nein\"
                },
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"no\"
                }
              ]
            },
            {
              \"dataValue\": \"1\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"ja\"
                },
```

```
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"yes\"
                }
              ]
            }
          ]
        },
        {
          \"questionType\": \"BOOLEAN\",
          \"required\": \"false\",
          \"dataElement\": \"de_13_3\",
          \"text\": {
            \"languageTextContainer\": [
              {
                \"languageCode\": \"de-de\",
                \"languageValue\": \"Volupta\"
              },
              {
                \"languageCode\": \"en-us\",
                \"languageValue\": \"Volupta\"
              }
            ]
          },
          \"itemList\" : [
            {
              \"dataValue\": \"0\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"nein\"
                },
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"no\"
                }
              ]
            },
            {
              \"dataValue\": \"1\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"ja\"
                },
```

```
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"yes\"
                }
              ]
            }
          ]
        },
        {
          \"questionType\": \"BOOLEAN\",
          \"required\": \"false\",
          \"dataElement\": \"de_13_4\",
          \"text\": {
            \"languageTextContainer\": [
              {
                \"languageCode\": \"de-de\",
                \"languageValue\": \"Gubergren\"
              },
              {
                \"languageCode\": \"en-us\",
                \"languageValue\": \"Gubergren\"
              }
            ]
          },
          \"itemList\" : [
            {
              \"dataValue\": \"0\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"nein\"
                },
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"no\"
                }
              ]
            },
            {
              \"dataValue\": \"1\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"ja\"
                },
```

```
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"yes\"
                }
              ]
            }
          ]
        },
        {
          \"questionType\": \"BOOLEAN\",
          \"required\": \"true\",
          \"dataElement\": \"de_13_5\",
          \"text\": {
            \"languageTextContainer\": [
              {
                \"languageCode\": \"de-de\",
                \"languageValue\": \"Justo\"
              },
              {
                \"languageCode\": \"en-us\",
                \"languageValue\": \"Justo\"
              }
            ]
          },
          \"itemList\" : [
            {
              \"dataValue\": \"0\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"nein\"
                },
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"no\"
                }
              ]
            },
            {
              \"dataValue\": \"1\",
              \"languageTextContainer\": [
                {
                  \"languageCode\": \"de-de\",
                  \"languageValue\": \"ja\"
                },
```

```
                {
                  \"languageCode\": \"en-us\",
                  \"languageValue\": \"yes\"
                }
              ]
            }
          ]
        },
        {
          \"questionType\": \"TEXT_INPUT\",
          \"required\": \"false\",
          \"dataElement\": \"de_13_6\",
          \"text\": {
            \"languageTextContainer\": [
              {
                \"languageCode\": \"de-de\",
                \"languageValue\": \"Sonstiges\"
              },
              {
                \"languageCode\": \"en-us\",
                \"languageValue\": \"Else\"
              }
            ]
          }
        }
      ]"
    }
  ]
}
```

Listing C.1: Seitenbeschreibung in JSON

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[9Pa14] *Draw 9-Patch*. http://developer.android.com/tools/help/draw9patch.html, letzter Zugriff: 26.01.2014

[agi14] AGILE.DE it: *Vorteile agiler Methoden, Was ist agile Softwareentwicklung?* http://www.it-agile.de/wissen/vorteile-agiler-methoden/, letzter Zugriff: 23.01.2014

[And14] *Android*. http://www.android.com/, letzter Zugriff: 26.01.2014

[Fie14] FIELDING, Roy T.: *RArchitectural Styles and the Design of Network-based Software Architectures*. http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm, letzter Zugriff: 24.01.2014

[Gar14] GARTNER: *Android erreicht fast 75 Prozent Marktanteil*. http://www.zdnet.de/88154889/gartner-android-erreicht-fast-75-prozent-marktanteil/, letzter Zugriff: 24.01.2014

[Jav14] *Java*. http://www.java.com/, letzter Zugriff: 26.01.2014

[KE13] KARL EILEBRECHT, Gernot S.: *Patterns kompakt, Entwurfsmuster für effektive Software-Entwicklung*. Springer Verlag; Auflage: 4, 2013. – ISBN 978–3–642–34717–7

[Kü12] KÜNNETH, Thomas: *Android 4: Apps entwickeln mit dem Android SDK: (Galileo Computing)*. Galileo Computing, 2012

[Mai12] MAIER, Fabian: *Entwicklung eines mobilen und Service getriebenen Workflow-Clients zur Unterstützung von evaluierten Studien der klinischen*

Psychologie am Beispiel der AristaFlow BPM Suite und Android, University of Ulm, Bachelor-Thesis, November 2012

[MG12] MIVO GEIS, Marcus H.: *Datenschutzrecht: Bundesdatenschutzgesetz, Informationsfreiheitsgesetz, Grundgesetz (Auszug), Verwaltungsverfahrensgesetz (Auszug)*. Deutscher Taschenbuch Verlag; Auflage: 4, 2012

[MJH09] MARC J. HADLEY, Sun Microsystems I.: *Web Application Description Language (WADL)*. `http://java.net/projects/wadl/sources/svn/content/trunk/www/wadl20090202.pdf?rev=328`, 2009

[Obj14] OBJECTAID: *ObjectAid UML Explorer*. `http://www.objectaid.net`, letzter Zugriff: 27.01.2014

[Off12] OFFERGELD, Michael: *Usability Engineering, Skript zur Vorlesung*. Institut für Medieninformatik, Universität Ulm, 2012

[Qui14] *QuickTabSurvey*. `http://www.quicktapsurvey.com/`, letzter Zugriff: 16.02.2014

[Sch12] SCHMID, Maximilian: *Technische Konzeption und Realisierung der Anwenderumgebung für ein generisches Fragebogensystem zur IT-gestützten Durchführung von evaluierten Studien der Klinischen Psychologie*, University of Ulm, Master-Thesis, Master-Thesis, September 2012

[Sch14] SCHERLE, Steffen: *Konzeption und Evaluierung einer domanenspezifischen Modellierungsumgebung fur prozessorientierte Fragebogen*, University of Ulm, Diploma, 2014

[SSP+13] SCHOBEL, Johannes ; SCHICKLER, Marc ; PRYSS, Rüdiger ; NIENHAUS, Hans ; REICHERT, Manfred: Using Vital Sensors in Mobile Healthcare Business Applications: Challenges, Examples, Lessons Learned. In: *9th Int'l Conference on Web Information Systems and Technologies (WEBIST 2013), Special Session on Business Apps*, 2013, S. 509–518

[Usa99] *The Usability Engineering Lifecycle*. DMorgan Kaufmann, 1999. – ISBN 978–1558605619

[Zel13] ZELLER, Patrick: *Konzeption und Realisierung eines Sensor-Frameworks für mobile Anwendungen und Integration von Sensorinformationen am Beispiel einer mobilen Fragebogen-Applikation*, University of Ulm, Master-Thesis, November 2013

Name: ALEXANDER REISSER Matrikelnummer: 428326

Erklärung

Ich erkläre, dass ich die Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Ulm, den .....

ALEXANDER REISSER