Bachelorarbeit
Interaktionsmechanismen für den Lernprozess einer mobilen e-Learning Applikation

Christian Kletzander[1] und Alexander Kögler[2]

[1] Weststrasse 23, 2273 Hohenau/March
kletzanderc@gmail.com
MatrNr.: 1125210
[2] Mitisgasse 8/5/2/164, 1140 Wien
alexander.koegler@chello.at
MatrNr.: 1125544

Betreuerin: Prof. Dr. Hilda Tellioglu
24.06.2014

Zusammenfassung Mobile Endgeräte nehmen in der heutigen Zeit einen hohen Stellenwert in allen Altersgruppen ein [12]. Diese Geräte finden heutzutage auch ihren Einsatz in Klassenräumen. Es gibt Applikationen, die von der Begleitung des Unterrichts bis zum spielerischen Lernen des Stoffgebietes reichen. Viele Benutzer erschließen jedoch nicht das volle Potential dieser Applikationen, weil sie mit dem Interaktionsdesign der Anwendung nicht umgehen können. Aufgrunddessen verlieren die Benutzer schnell den Spaß und die Motivation am Benutzen der Lösung. Diese Bachelorarbeit benutzt als Grundlage die Erkenntnisse von Alexander von Franqué [29], welche in die Apple iOS-Applikation "UML-Quiz" eingeflossen sind und versucht dort anzusetzen, wo das Interaktionsmenü Schwächen aufzeigt. Die grundlegende Analyse dieser Anwendung aber auch die Evaluierung der Ergebnisse mit potenziellen Endbenutzern unterstützen den Gestaltungs- und Verbesserungsprozess dieser e-Learning Applikation. Das Ziel liegt darin eine e-Learning Anwendung zu erstellen, die an die Bedürfnisse der BenutzerInnen angepasst ist, um ein optimales Lernergebnis mithilfe der Software erzielen zu können.

Inhaltsverzeichnis

1 Abgrenzung - State of the Art 1
2 Theorie 1
2.1 Erläuterung der Unterschiede von Interaktionsmöglichkeiten zwischen Android und iOS 2
2.2 Analyse der App Navigation in Android 2
2.2.1 BackButton, der Zurück-Knopf in Android 2
2.2.2 Bidirektionale Navigation 4
2.2.3 Accessibility unter Android 5
3 Analyse 5
3.1 Theoretische Designansätze 6
3.2 Methoden der Usability Evaluierung 8
3.3 Partizipative Systemgestaltung 9
3.4 Nutzergruppe 9
3.5 Analyse der iOS - Bestandsapplikation 11
3.6 Analyse theoretischer Designansätze 16
3.7 Analyse von e-Learning Quiz Applikationen 18
4 Praxis 21
4.1 Analyseerkenntnisse 21
4.2 Entwurf 23
4.3 Iterativer Entwicklungsprozess 25
4.3.1 Darstellung der Fragen 26
4.3.2 Farbliche Gestaltung 29
4.3.3 Grafische Trennung zwischen Fragetext und Antwort 30
4.3.4 Hilfestellungen für die NutzerInnen 30
4.3.5 Accessibility 34
5 Technische Dokumentation 36
5.1 Programmierdokumentation 36
5.2 Anforderungsdokumentation 37
5.3 Installationsdokumentation 37
5.4 Anwenderhandbuch 37
5.5 Architekturbeschreibung 41
5.5.1 Datenbank - Schicht 41
5.5.2 Business - Schicht 42
5.5.3 GUI - Schicht 43
5.5.4 XML - Schnittstelle 43
5.5.5 Arbeitspakete der ersten Iteration 45
6 Analyse und Erkenntnisse 45
6.1 Konklusio des iterativen Entwicklungsprozesses 46
6.1.1 Meinung der Entwickler 46
6.1.2 Meinung der Testbenutzer 48
7 Anhang 51

7.1 Protokolle 51
7.1.1 Erste User Evaluierung 51
7.2 XML Beispielcode 55
Literatur 56

1 Abgrenzung - State of the Art

Derzeit werden in der Lehrveranstaltung, für die jene Applikation entwickelt wird, ein Online-Testsystem basierend auf der Moodle-Lernplatform und eine davon portierte Anwendung für das Apple Betriebssystem iOS angeboten und intensiv genutzt. Es werden Multiple-Choice Fragen mit Illustrationen dem Benutzer geboten, die der Reihe nach zu beantworten sind. Am Ende vom Quiz können die Fragen auf Richtigkeit überprüft werden. Außerdem werden Hinweise und Erklärungen ebenfalls individuell zur Frage angezeigt.

Während sich die Bachelorarbeit [29] von Alexander von Franqué eingehend damit beschäftigte, den zuvor in Moodle bereitgestelllten Content betreffend beschränkter Interaktionsmöglichkeiten auf mobilen Geräten mit Gestenerkennung bereitzustellen, fokussiert diese Arbeit darauf die Usability und Effizienz der Wissensabsorption durch Adaptierung der mobilen Anwendung für Android zu steigern und die Auswirkungen mittels Testläufen bei Benutzergruppen zu evaluieren. So wurde in der Arbeit von Alexander von Franqué [29] Wert auf das Layout und den Umfang des Contents gelegt, auf dem Resultat und hervorgegangenen Empfehlungen basiert diese Arbeit.

Usability und Interface Design sind ein Schwerpunkt in dieser Bachelorarbeit. Verschiedene Layouts sollen für Android entworfen (Kletzander) und getestet werden, sowie Gestensteuerung (Kögler) umfassender eingesetzt werden. Es wird auch auf die Farbgestaltung (Kögler) eingegangen, inwiefern diese dazu beitragen kann, die mobile Anwendung und die Nutzung für Benutzer mit Seh- und Farbschwäche zu verbessern. Die Interaktion mit der von Android angebotenen Sprachausgabe wird ebenfalls untersucht (Kögler) und ob diese Personen mit eingeschränkter Sehfähigkeit unterstützen kann.

Die von Android zur Verfügung stehende Technologie (Kletzander) und Interaktion mit den Benutzern ist ein weiteres Thema in dieser Arbeit. Untersuchungsergebnisse für Lern- und Quizapplikationen auf mobilen Endgeräten werden erörtert und für die zu entwickelnde Android Lösung genutzt. Gibt es Features die Android im Gegensatz zu iOS bereitstellt und vice versa? Welche Anpassungen müssen notwendigerweise im Bezug zur bereits bestehenden iOS-Applikation und dem Content für die Portierung auf Android gemacht werden? Kann die Aktualität und Wartbarkeit der Software verbessert werden, vor allem betreffend zukünftiger Lerninhalte?

Die Evaluierung der Lösung mit den Nutzern stellt schließlich das Untersuchungsergebnis der Bachelorarbeit dar.

2 Theorie

Die Betriebssysteme iOS und Android sind von ihrer Architektur ausgehend sehr unterschiedlich. Die Interaktions- und Designmöglichkeiten unterscheiden

sich sehr stark voneinander. Im Folgenden werden die wichtigsten Unterschiede näher erläutert.

2.1 Erläuterung der Unterschiede von Interaktionsmöglichkeiten zwischen Android und iOS

Die grundlegende Applikation basiert auf dem Betriebssystem iOS von Apple. Die Architektur zu Android unterscheidet sich grundlegend darin, dass iOS keinen Zurück-Knopf besitzt [7]. Viele iOS-NutzerInnen vermissen diese Art von Interaktion nicht, sobald man jedoch Android-NutzerInnen fragt, wie sie sich das Arbeiten ohne Zurück-Knopf vorstellen würden, so ist dieser Zusatz-Knopf unabdingbar [27]. Diese Erkenntnis geht auch aus einer Befragung unserer potentiellen Endnutzergruppe heraus.

Die Individualität des Android Betriebssystems ist ebenfalls ein großer Vorteil, wenn es um die Interaktion innerhalb einer Anwendung geht. Unter iOS sind grundlegende Elemente wie Textfelder, Check-Boxen oder Select-Boxen einheitlich strukturiert und dargestellt. Unter Android besteht die Möglichkeit das Layout jedes einzelnen Elements mit seinen eigenen Vorstellungen zu überschreiben. Dies ermöglicht es Interaktionsmöglichkeiten zu implementieren, die abhängig von bestimmten Farbimpulsen sind, sowie wichtigere Elemente in den Vordergrund und Unwichtiges in den Hintergrund zu stellen. Das Zeichnen der grafischen Benutzeroberfläche ermöglicht somit Individualität innerhalb der Applikation und bringt großes Potential mit sich, welches mit den Benutzern ausgeschöpft wird.

Ein Element, welches man unter iOS vergeblich sucht, ist die Möglichkeit ein Menü zu schaffen, welches unterschiedlichste Funktionen innerhalb der Lösung zur Verfügung stellt, ohne die aktuelle Seite zu wechseln. Dadurch bekommt man die Möglichkeit ein einheitliches Menü auf jeder Seite zu platzieren, welches dem Benutzer schnell einleuchtend ist und eine Navigation zwischen den einzelnen Seiten oder Hilfedialogen zur Verfügung stellen kann.

2.2 Analyse der App Navigation in Android

Das UML-Quiz sollte, um Design und Usability Prinzipien gerecht zu werden, nicht eins zu eins kopiert und auf Android portiert werden, sondern vielmehr wird als Ziel angesehen, die negativen Aspekte der iOS Anwendung zu verbessern und die portierte Version an die Gegebenheiten eines Android Systems anzupassen, um die Erwartungskonformität der AnwenderInnen zu erfüllen [25].

2.2.1 BackButton, der Zurück-Knopf in Android Im Gegensatz zu iOS bietet Android in jeder Anwendung, innerhalb als auch untereinander, den Back-Button an. Dieser dient ausschließlich dazu im Kontext rückwärts, aber keinesfalls vorwärts, zu navigieren. Dies hat mehrere Gründe, denn es sollte eine Be-

nutzerinteraktion auf Mobilgeräten einfach gestaltet sein und mit wenigen Interaktionen durchgeführt werden können. Speziell bedeutet dies, dass Interaktionen vorwärts durch wiederholtes Ausführen bisheriger Schritte schnell durchgeführt werden können.

Es sollte dem Benutzer einheitlich in jeder Software die Möglichkeit geboten werden, den Kontext mit einem Schritt rückwärts zu navigieren. Einerseits in der Anwendung selbst, andererseits um untereinander Anwendungen zu wechseln, um zu einer zuvor unterbrochenen Tätigkeit zurückzukehren. Als Beispiel für eine Interaktion innerhalb einer Anwendung kann hierfür die Navigation in einem Webbrowser genannt werden, wobei der Benutzerin jederzeit die Möglichkeit gegeben werden muss, zur vorhergehenden Webseite wechseln zu können.

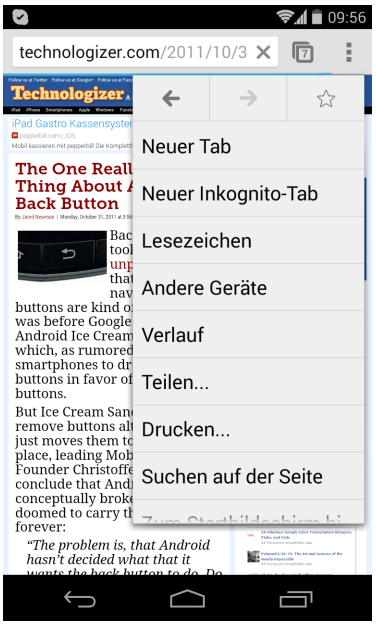

Abbildung 1. Zurück-Knopf im Android Chrome-Browser

Als Szenario für Kontextwechsel zwischen Programmen könnte ein Musikabspielprogramm herhalten. Die Benutzerin befindet sich in einer anderen Anwendung, die Musik hört auf zu spielen. Die Anwenderin wechselt in die Musiksoftware, wählt ein neues Lied aus, betätigt schließlich den BackButton, um in das vorhergehende Programm zu wechseln.

Das Fehlen des BackButtons auf iOS Geräten wird von einigen Nutzern kritisiert, denn so muss der Softwareentwickler selbst dafür Sorge tragen, einen benötigten BackButton zu erstellen und mit Funktionalität zu verknüpfen.

Im Gegensatz dazu wird die zum Teil existierende inkonsistente Funktionalität des BackButton in Frage gestellt. Dabei muss der Android App-BackStack von entwickelnden Personen passend angelegt werden. Jede Anwendung kann mehrere Activities beinhalten. Eine Anwendung kann eine andere Anwendung starten. Jede Activity wird am BackStack abgelegt, sofern Android nicht von der App-Konfiguration davon abgehalten wird. Betätigt man den BackButton, wird die zuvor am BackStack abgelegte Aktivity angezeigt. Wird auf diesem jede Activity abgelegt, führt dies mitunter dazu, dass der Anwender eine Vielzahl an Activities angezeigt bekommt, die für den weiteren Verlauf der Interaktion nicht von Bedeutung sind, sobald der BackButton, mit Beispielsweise dem Ziel das Hauptmenü zu erreichen, betätigt wird [27].

2.2.2 Bidirektionale Navigation Für das UML-Quiz ist es unerlässlich die Fragen vorwärts und rückwärts zu navigieren, bis das Quiz aufgelöst wird. Dies geschieht in der iOS Anwendung mittels horizontaler Swipe-Geste. Dieses Bedienungsparadigma ist auch für Android zu übernehmen. Explizite Vorwärts- und Rückwärtsknöpfe sind zu vermeiden. Als Gegenbeispiel für dieses Paradigma kann der Chrome Browser für Android angesehen werden. Dieser beinhaltet Vor- und Rückwärtsnavigation wie die Desktopversion im Menü. Dies ist mit der gewohnten Bedienung am Desktop argumentierbar. Auch die Vorwärtsnavigation wird erleichtert, da es mitunter nicht schnell möglich ist, auf einer Webseite mit viel Text den erneut anzuwählenden Link zu finden. Der Einsatz und Nutzen dieser zusätzlichen Navigationsmöglichkeit als Ergänzung zur Swipe-Geste müsste nach Rücksprache mit Benutzern evaluiert werden.

Geplant ist das App-Icon der Actionbar für die Rückwärtsnavigation innerhalb der App zu nutzen. Beispiel für eine Umsetzung dieser Navigationsmöglichkeit ist die ntv-App aus dem Google Play Store. Auf oberster Ebene wird beim Betätigen des App-Icons ein Schnellauswahl-Menü angezeigt. Der von Android gebotene BackButton wird dieselbe Funktionalität bieten. Es ist nicht geplant die Rückwärtsnavigation für das Ansteuern der Fragen zu nutzen, sondern um zu verschiedenen Aktivitäten (bspw. das Hauptmenü) des UML-Quiz zu wechseln. Es ist geplant ausschließlich das Hauptmenü und den Quizmodus mit dem BackButton zu erreichen. Zum einen wird dadurch sichergestellt, dass in drei Schritten ein Wechsel zur zuletzt aufgerufenen App ermöglicht wird, zum anderen, dass ein neues Quiz oder der Wechsel zu einem anderen, mit zwei Schritten zurück zu bewerkstelligen ist.

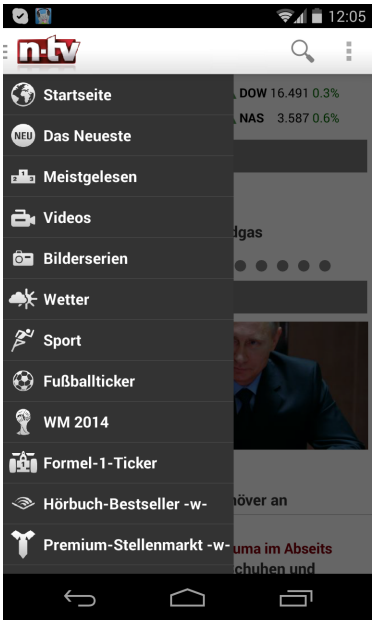

Abbildung 2. ActionBar in der n-tv Applikation

2.2.3 Accessibility unter Android Betreffend Accessibility gibt es auf Android Systemen das Text-To-Speech (Talk-Back) [13]. Die Programmiererin/Der Programmierer muss dabei sicherstellen, dass zu jedem Steuerelement ein "beschreibender Text" vorhanden ist. Hat der Anwender/die Anwenderin Explore-by-Touch aktiviert, so wird dieser Text von Android vorgelesen, sofern ein Steuerelement fokussiert wird. Explore-by-Touch bietet die Möglichkeit für blinde oder stark sehbeeinträchtigte Personen, die Touch Oberfläche mit dem Finger abzutasten, ohne unbeabsichtigt dabei Eingaben zu tätigen. Das Feature ist ab Android Version 4.0 verfügbar und dessen Nutzbarkeit daher eingeschränkt [13].

3 Analyse

In der Analysephase beschäftigt man sich stark mit der Analyse der Bestandsapplikation, der Analyse von vorhandenen Design-Patterns im mobilen Bereich, sowie der Analyse von bereits vorhandenen Anwendungen am Markt und dieser Umsetzung. Zur Unterstützung des Analyseprozesses und zur effektiven Verfolgung des Ziels, der Gestaltung und Findung von Interaktionsmöglichkeiten innerhalb einer e-Learning Applikation, wird eine ausgewählte Gruppe von möglichen Benutzern in diesen Prozess eingebunden. Es kommt der Prozess des partizipativen Designs [25] zum Einsatz, der die Nutzer in den Produktentwicklungsprozess einbindet. Besonderes Augenmerk wird auf die Analyse-, Entwurfs- und

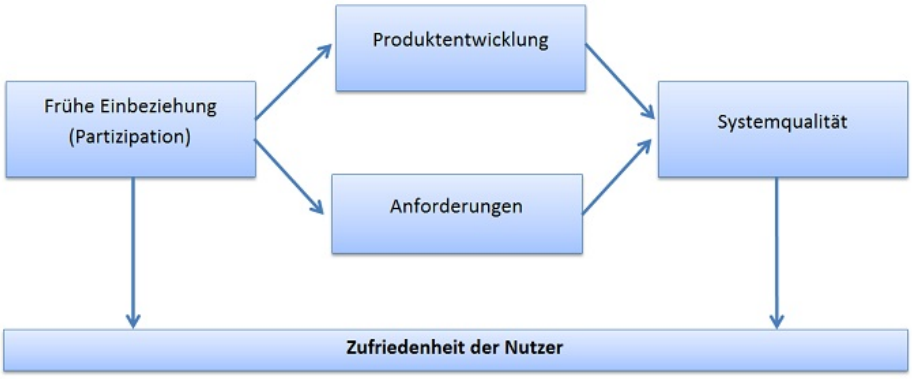

Abbildung 3. Auswirkungen partizipativen Designs [28]

Evaluierungsphase gelegt. In dieser Phase sollen die Nutzer besonders intensiv eingebunden werden.

3.1 Theoretische Designansätze

Einhergehend mit einer kritischen Begutachtung der iOS Lösung wurden Verbesserungsvorschläge erarbeitet und Theorien gesucht, die diese auch manifestieren konnten. Dabei wurde sowohl auf die menschliche Verarbeitungsfähigkeit, als auch im speziellen auf die Nutzung eines mobilen Geräts, eingegangen.

Bereits das Lesen von Texten (speziell die Fragen des Quiz), aber auch die Verständlichkeit von Dialogen, scheinen sie noch so trivial, erfordert, dass diese leicht vom Auge erfasst werden können, damit diese im Gedanken schnell, richtig, ohne Verwirrung und mit einem Lernerfolg verarbeitet werden können. Die temporale physische Grenze für das Lesen ist zehnfach höher als die tatsächliche Auffassungsfähigkeit von technischen Texten und Formeln, die für die kognitive Verarbeitung notwendig ist. Die Geschwindigkeit reduziert sich bis auf 75 Wörter in der Minute [2]. Beim Lesen von Texten auf Bildschirmen verringert sich die Geschwindigkeit gar nochmals um ein Drittel [14]. Konkret bedeutet dies, dass für eine mobile e-Learning Anwendung die Länge von Texten (Fragestellungen, Antworten, Erklärungen) bedacht werden muss. Insbesonders wenn das Ziel, ist diese kurzzeitig zu benutzen. Die Textlänge ist daher ein wichtiges Kriterium für die Nutzung und Akzeptanz durch die BenutzerInnen.

Für das Erreichen des Lernzieles trägt neben der Formulierung auch die Gestaltung der Benutzeroberfläche bei. Mehrere unterschiedliche, doch zueinander passende Schriftarten können dem Gehirn helfen Informationen zu differenzieren. Empfohlen wird, nicht mehr als drei verschiedene Schriftarten zu verwenden, da ansonsten die Benutzerin/der Benutzer verwirrt und abgelenkt wird. Eine räumliche Anordnung verschiedener Gestaltungelemente kann der Unterstützung

beim Lernprozess ebenfalls Rechnung tragen [15].

Grafische Elemente (Emoticons, Symbole) sollen hierbei nicht nur eingesetzt werden, um das Lernen zu unterstützen, sondern vor allem auch für Nutzer mit Farbschwäche bei der Unterscheidung behilflich sein. Besonders die Rot-Grün-Farbschwäche muss beim Design berücksichtigt werden, da diese bei Männern besonders häufig auftritt. Obwohl mit Förderungsprogrammen versucht wird, mehr Frauen für das Informatikstudium zu gewinnen, bleibt der Anteil an Männern weiterhin sehr hoch. Dies bedeutet, dass eine große Anzahl an Studenten, mit dem derzeitigen Design Schwierigkeiten haben wird, wenn ungefähr acht Prozent aller Männer eine Farbschwäche haben.

Die Nutzung von Emoticons wirkt sich positiv auf die zwischenmenschliche Kommunikation und den Unterhaltungswert von Kurzmitteilungen aus [16]. Im Gegensatz zu abstrakten grafischen Symbolen deren Bedeutung erlernt werden muss, sind soziokulturelle Erfahrungen, der ersten Lebensmonate, für die korrekte Assoziation von Emoticons verantwortlich [2]. Das Lachen wird auf Evolutions-Hypothesen zurückgeführt und von Kleinkindern bereits sechs Wochen nach der Geburt aktiv im Umgang mit Mitmenschen genutzt [19]. Ein lachend oder traurig blickendes Emoticon wird dennoch nicht in allen Kulturen gleich interpretiert [22]. Der Einsatz von Icons, Emoticons, deren Nutzen, sowie passende Interpretation durch den Anwender ist daher hauptsächlich durch Usability-Tests sicherzustellen [26].

Am Rande betrachtet und für das UML-Quiz weniger relevant sind auch die kognitiven Eigenheiten des Menschen betreffend der Ersetzung alter Software mit einer Neuentwicklung. Obwohl die Neuentwicklung besser ist, muss diese auch nicht in Folge von den Nutzern anerkannt und verwendet werden. Das "magische Denken" sorgt dafür, dass "neue menthale Modelle so konstruiert werden, dass sie zu bereits Erlerntem passen" [25]. Als ein gutes Beispiel dafür kann die vielfache Kritik bei der Veröffentlichung der letzten großen Betriebssysteme von Microsoft (Windows XP, Vista, 7, 8) hergenommen werden. Geschuldet dem hohen Nutzungsgrad wurde XP am Anfang für zu bunt und aufgrund der Menüführung in der Systemsteuerung umständlich angesehen, ist es mit zwölf Jahren Herstellerunterstützung eines der am längsten gewarteten Betriebssysteme der großen Masse.

Je länger eine Software verwendet wird, umso mehr gewöhnen sich AnwenderInnen an die positiven Aspekte. Neuerungen werden daher von vornherein eher abgelehnt. Der Begriff der "kognitiven Dissonanz" bezeichnet die psychologische Eigenheit, dass Gedankenmodelle soweit angepasst werden, bis diese ins vertraute Weltbild passen, um unangenehme Situationen zu verdrängen [25]. Damit eine Neuentwicklung auch von den Nutzern akzeptiert wird, sind AnwenderInnen nicht nur in der Softwareentwicklung, sondern auch beim Bereitstellen und in der Anfangsphase des Betriebs zu betreuen.

Die Evaluierung der iOS Software, anhand dieser Kriterien, ist in Kapitel 3.5 dokumentiert.

3.2 Methoden der Usability Evaluierung

Theoretische Überlegungen und Grundlagen auf physiologischen, psychologischen Aufnahmefähigkeiten existieren, trotzdem kann meist kein komplexeres System einzig und allein damit gestaltet werden. Schwere Probleme sind in vielerlei Hinsicht nicht offensichtlich und somit auch nicht so leicht zu finden wie einfache Usability Probleme. Falsch gewählte Formatierungen sind bald gefunden, aber wirken sich nicht zwingend negativ auf die Funktionalität aus. Selten verwendete Funktionen in Verbindung mit komplex verknüpften Hintergründen und Spezialfällen, die das Programm in einem gewissen Zustand zum Absturz bringen können, werden oftmals nicht mit automatisierten Tests gefunden werden. Einerseits durch die Höhe der Komplexität, wodurch in absehbarer Zeit nicht alle möglichen Fälle getestet werden können, andererseits begründet auf der Unlösbarkeit des Halteproblems [25].

Es lässt sich daher herleiten, dass eine ausreichend stabile Funktionsweise eines Programms dann sichergestellt werden kann, wenn eine Auswahl der späteren BenutzerInnen beim Design und Testprozess involviert werden. Hierfür gibt es eine Reihe ausgearbeiteter Normen (DIN EN ISO 13407, DIN EN ISO 14915, ...) die ein "aktives Einbeziehen der Nutzer" aus verschiedenen Anwenderdomänen unabhängig von deren Position in der Hierarchie, Kenntnis der Technologie, oder Erfahrung vorsehen [25].

Ein stabiles System muss auch auf etwaige Fehler der BenutzerInnen Rücksicht nehmen. Nicht nur fehlerhafte Bedienung und Eingaben in einem Kontext sind auszuschließen, sondern gerade im Bezug zur Softwarenutzung auf einem Mobiltelefon ist davon auszugehen, dass die Anwenderin/der Anwender diese nur nebenbei und unkonzentriert bedient. Eine der häufigsten Fehlerursachen stellen "Versäumnisse durch Unterbrechung" dar wie das Telefon klingelt, Nachrichten werden in sozialen Clients fortwährend gelesen und versendet, während dem Umstieg zwischen Verkehrsmitteln unterbrechen Anwender am Mobiltelefon den aktuellen Interaktionsprozess.

Die von [25] erwähnte History-Funktion ist auf derzeitig verfügbaren Mobilgeräten von Werk aus vorhanden. Ein Wechsel zwischen verschiedenen Anwendungen, anhand der zeitlichen Aufrufsreihenfolge, ist möglich. Hingegen muss bei der individuellen Softwareentwicklung ein Konzept erarbeitet werden, in dem ausgearbeitet wird mit welcher Art von Rückmeldung die Anwenderin/-der Anwender erfährt, welche Schritte im Kontext bereits ausgeführt wurden und welche noch ausstehend sind. Benutzereingaben, Einstellungen und derzeitige Position in einem Kontext müssen von mobiler Software bei Unterbrechung beibehalten werden [9].

3.3 Partizipative Systemgestaltung

Unter partizipativer Systemgestaltung wird die Einbeziehung der späteren Nutzer in den Entwicklungsprozess verstanden. Sarodnick hat in seinem Buch [25] mehrere Möglichkeiten zusammengetragen, wie AnwenderInnen in den Systementwicklungsprozess eingebunden werden können.

Für die Entwicklung des UML-Quiz kamen aus Zeit-, Kosten- und Aufwandsgründen nur die folgenden in Frage. Eine "Teilnehmende Beobachtung" bei der das Nutzerverhalten durch Beobachtung und aktive Befragung erhoben wurde. "Allgemeine Interviews, sowie Fokusgruppen" [18] bei denen GUI Skizzen erarbeitet und diskutiert wurden. Ferner kann auch die "Zukunftswerkstätte" genannt werden, da mit den Anwendern überlegt wurde, welche fehlenden Features noch eingebaut werden könnten oder welche Funktion andere Software bietet, die in das UML-Quiz übernommen werden sollten.

Um das Accessibility Feature Text-To-Speech auszutesten, wurden einem Propanden die Augen verbunden, mit einem Beobachter gemeinsam die Anwendung erkundet. Für die Entwicklung war das erste Treffen mit Anwendern Ausschlag gebend, bei dem erste GUI Entwürfe kritisiert, Verbesserungvorschläge erteilt wurden.

3.4 Nutzergruppe

Um den Produktentwicklungsprozess möglichst benutzerorientiert gestalten zu können, wurde kurzzeitig eine Nutzergruppe gebildet. Diese unterstützen den Analyse-, Entwurfs- und Evaluierungsprozess der Applikation. Die Nutzergruppe teilt sich in verschiedene Untergruppen auf. Diese bestehen aus potentiellen Anwendern (Studenten der TU Wien), Nutzer in einem Alter zwischen 30 und 50 Jahren, Nutzer, die schon ähnliche Systeme auf ihren Geräten betreiben und Nutzer, die möglichst einfach und schnell mit ihrem mobilen Endgerät zu einem definierten Ziel kommen möchten. Durch die Mischung der unterschiedlichen Interessen wird versucht ein breitgefächertes Ergebnis zu erzielen, welches die Interaktion mit der Anwendung einfach macht und den Lernerfolg in den Vordergrund stellt.

Für die Testpersonen war das UML-Quiz unbekannt, hingegen waren aber Erfahrungen mit anderen ähnlichen Systemen (Anki, Facebook-Quiz, Moodle) bereits vorhanden. Unter den Testern waren Studenten verschiedener Fachrichtungen und Universitäten Wiens, nebst Werktätigen ohne höheren Schulabschluss. Die getätigte Auswahl an unterschiedlichen Nutzern kann somit als dem Ideal für die partizipative Entwicklung (siehe Kapitel 3.3) angesehen werden [20]. Auch die Anzahl von bis zu fünf Personen entspricht den Empfehlungen von Sarodnick [25].

Die NutzerInnen werden in Anlehnung an DIN EN ISO 13407 in den unterschiedlichsten Phasen des Projekts eingegliedert. Es ist nicht notwendig, dass diese Teams allzu groß sind und für die Gesamtdauer der Softwareentwicklung bestehen bleiben. Durch eine frühzeitige Einbindung der AnwenderInnen im Sinne einer partizipativen Gestaltung kann ein besonders positiver Einfluss auf die Nutzerzufriedenheit festgestellt werden [25].

Eine der ersten Phasen ist die Analysephase. In dieser Phase wird einem Teil der Nutzergruppe die bereits bestehende iOS Applikation vorgeführt. Die Nutzer können die Anwendung auf einem zur Verfügung gestellten Gerät ausprobieren und ihre Kritik äußern. In dieser Phase versucht man grobe Interaktionsfehler ausfindig zu machen und ein erstes Bild von der Bestandsapplikation zu bekommen. Während der haptischen Exploration der Lösung ist es ebenfalls nötig, die NutzerInnen genau zu beobachten, denn ein verzögertes Reagieren auf bestimmten Seiten der Lösung bedeutet den Verlust des Verständnisses über diese.

In der nächsten Phase haben die NutzerInnen die Möglichkeit ihre Vorstellungen und Wünsche an das Design zu stellen. Welche Seiten der Applikation sollten grundlegend überarbeitet werden und welche waren problemlos benutzbar. In dieser Phase wurden die zuvor getrennten Gruppen wieder zusammengeführt. Das bedeutet, dass ein Teil der Gruppe ein gewisses Vorwissen über das Aussehen der Bestandsapplikation hat und der Rest ihren Gedankenspielraum freien Lauf lassen kann, was die Gestaltung der individuellen Seiten der Anwendung betrifft.

Zum Abschluss der Analysephase werden der Nutzergruppe Lösungen vorgelegt, die ebenso aus dem e-Learning Bereich sind. Das Ziel dieser Phase ist es, Erkenntnisse über die Umsetzung im e-Learning Bereich zu sammeln, sowie herauszufinden, welche Bereiche dieser Lösungen besonders gut umgesetzt wurden und welche bei den Nutzern durchgefallen sind. Dies ermöglicht es aus bereits umgesetzten Interaktionsdesigns zu lernen und diese entsprechend in die e-Learning Applikation einzubauen.

Mit den gewonnenen Erkenntnissen der Analysephase werden Entwürfe (Mockups) erstellt. Besonderes Augenmerk wird hierbei auf das Design der Fragen gelegt, welches den wichtigsten Punkt der gesamten Applikation darstellt. Für diesen Punkt sollen drei Design-Vorschläge gefunden werden, die anschließend der Nutzergruppe zur Auswahl vorgelegt werden sollen. Die Nutzergruppe entscheidet, welches der erstellten Designs in der Applikation umgesetzt werden soll. Um dies zu entscheiden, sollen alle gesammelten Kriterien und Eindrücke der NutzerInnen in ein Design fließen.

In der abschließenden Evaluierungsphase wird der Nutzergruppe die entstandene Applikation vorgelegt. Das Ziel dieser Phase ist es, nach der ersten Entwicklungsiteration Kritik an den Interaktionsmöglichkeiten innerhalb der Applikation zu

äußern.

Durch diesen Prozess gelingt es ein Produkt zu entwerfen, welches auf die Zufriedenheit der Nutzer zielgerichtet ist und mit deren Akzeptanz für die Verwendung zu rechnen ist.

3.5 Analyse der iOS - Bestandsapplikation

Der erste Analyseprozess beschäftigt sich mit der bereits existenten iOS - Bestandsapplikation. Diese Applikation wurde anfänglich mit dem Ziel gestaltet, Fragen aus einer Online-Lernplattform zu importieren und diese als Quiz in der Anwendung bereitzustellen. Das Hauptaugenmerk wurde auf die Umsetzung der Anforderung gerichtet und hat das Interaktionsdesign in den Hintergrund gestellt. In der Bachelorarbeit [29] zur iOS - Applikation gibt es nur wenige Nutzertests bezüglich des gewählten Designs.

Im ersten Schritt wurde ein Aktivitäts-Diagramm erzeugt. Das Android-System basiert auf sogenannten Activities. Eine Seite der Applikation wird dabei als Aktivität dargestellt. Die erste Analyse beschäftigt sich damit, die Aktivitäten aus der iOS - Applikation herauszukristalisieren und ein Ablaufdiagramm zu erzeugen.

Basierend auf diesem UML-Ablaufdiagramm müssen die einzelnen Aktivitäten erzeugt werden und für die einzelnen Aktivitäten ein Interaktionsdesign gefunden werden.

Im nächsten Schritt wurde das Design der iOS - Applikation analysiert. Auf den ersten Blick konnte man feststellen, dass das grundlegende Design gut durchdacht ist. Es wird ein einheitlicher Designfaden durch die Applikation gezogen.

Wenn man die Funktionalität betrachtet, so läuft man leicht in eine Art Verwirrung, denn wenn man auf bestehende Knöpfe drückt, so erwartet man dasselbe Ergebnis wie beim ersten Mal, jedoch ändern sich innerhalb der Lösung die Ziele der Knöpfe dynamisch, je nachdem in welchem Status sich das Quiz gerade befindet. Bestimmte Seiten der Anwendung erscheinen nur in bestimmten Zuständen. Es fehlt ein einheitlicher Weg, den man durchläuft, wenn man ein Quiz starten möchte. Diese verschiedenen Wege, bis zum Start von einem Quiz, sind auch im Ablaufdiagramm ersichtlich.

Die Hauptaktivität der gesamten Applikation, das Interaktionsdesign der Darstellung einer Frage und ihrer Antworten, ist besonders auffällig. Hier wurde eine horizontal geteilte Darstellung der Frage gewählt.

Das mobile Endgerät kann nur im horizontalen Modus gehalten werden. Das bedeutet, dass man immer beide Hände braucht, um das Gerät bedienen zu können. Dies bringt Nachteile mit sich, denn wenn man unterwegs ist und ein

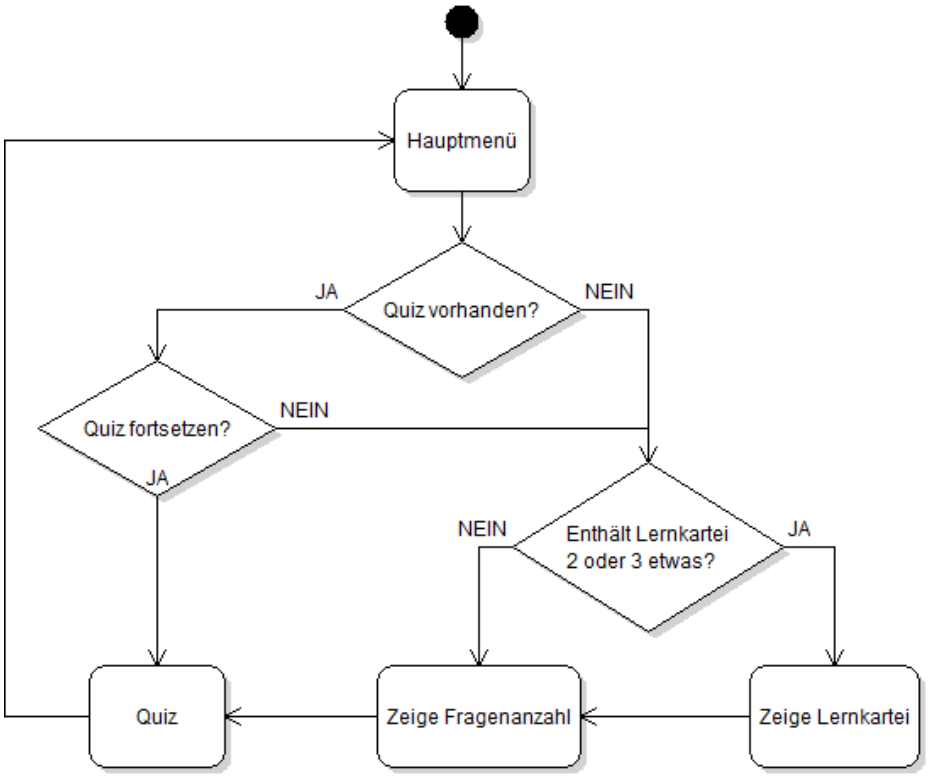

Abbildung 4. UML-Ablaufdiagramm der iOS - Bestandsapplikation

schnelles Quiz machen möchte und nur eine Hand frei hat, so ist man mit dieser Art von Interaktionsdesign maßlos überfordert. Das bedeutet, dass die Applikation nur noch eingeschränkt benutzbar ist. Außerdem schränkt diese Art von Aufteilung Linkshänder immens ein. Diese agieren hauptsächlich mit der linken Hand. Das bedeutet, dass die linke Hand über die Frage greifen muss, um die Antworten antippen zu können. Dadurch wird die Frage komplett verdeckt. Die NutzerInnen haben einstimmig ein Interaktionsdesign gefordert, welches sowohl horizontal - mit zwei Händen - als auch vertikal - mit einer Hand - bedienbar ist.

Ein interessanter Aspekt ist die Aufteilung der Darstellung. In der linken Hälfte des Bildschirms befindet sich die Frage und in der rechten Hälfte die dazugehörigen Antwortmöglichkeiten. Beide Bereiche stellen hierbei horizontales Scrollen zur Verfügung, falls die Bereiche mehr Platz einnehmen sollten. Der Bachelorarbeit [29] ist zu entnehmen, dass zu dieser Oberfläche verschiedenste Tests gemacht wurden, wie man die Aufteilung besser darstellen kann. Der wichtigste Aspekt der BenutzerInnen bestand darin, dass die Frage dauerhaft ersichtlich

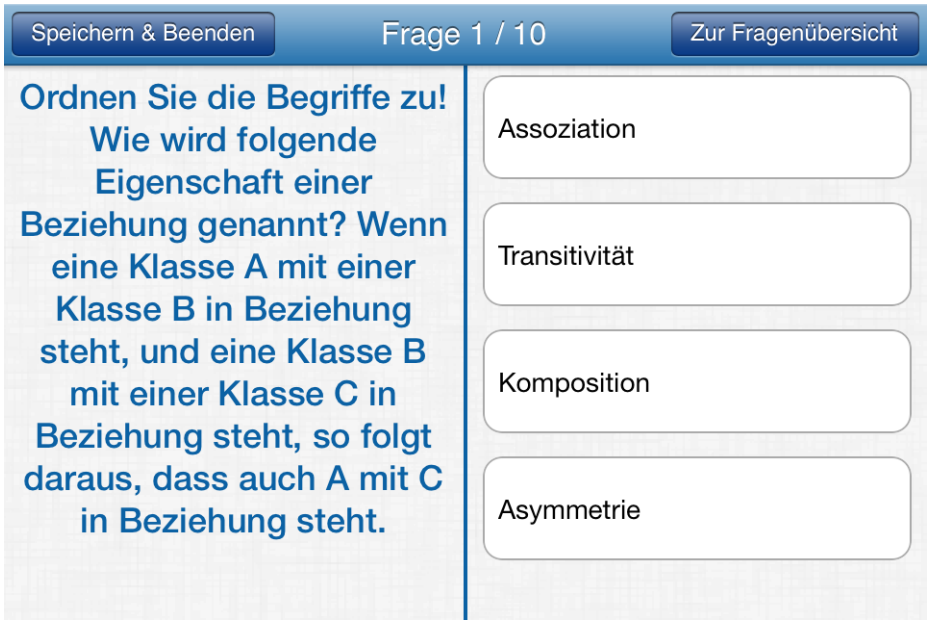

Abbildung 5. Fragenansicht der iOS - Bestandsapplikation

sein soll, um die Antwortmöglichkeiten besser zuteilen zu können.

Dieser Erkenntnis widersprechen jedoch unsere Nutzer. Die Analyse durch unsere Nutzergruppe ergab, dass es für sie überhaupt nicht im Vordergrund steht, dass die Frage immer ersichtlich ist, denn der Bildschirm stellt nur eine begrenzte Darstellungsmöglichkeit zur Verfügung. Durch das permanente Einblenden der Frage wird immer ein großer Teil des Bildschirms gefüllt und nimmt viel Platz weg für einen weiteren wesentlichen Bereich.

Außerdem wurde das grundlegende Layout dieser Aktivität in Frage gestellt. Alle NutzerInnen können sich ein Design wie dieses nicht vorstellen. Es liegt in der Natur von einem Quiz, dass die Frage über den Antwortmöglichkeiten dargestellt wird. Das bedeutet, dass das grundlegende Interaktionsdesign der Bestandsapplikation in Frage gestellt wurde. Man hat sich trotzdem damit befasst die Vorteile dieser Layoutierung in den Vordergrund zu stellen, jedoch konnte man keine wirklichen Vorteile für die Benutzerin/den Benutzer erkennen. Der einzige wesentliche Vorteil für die Funktionalität der Applikation war der, dass die Fragen und Antworten eine ungeahnte Länge annehmen könnten und somit den Rahmen des Bildschirms sprengen könnten, jedoch haben die Nutzer darauf vertraut, dass für dieses mögliche Problem eine Lösung gefunden wird.

Außerdem wurde die Wahl der Symbole innerhalb des Fragen-Layouts kritisiert.

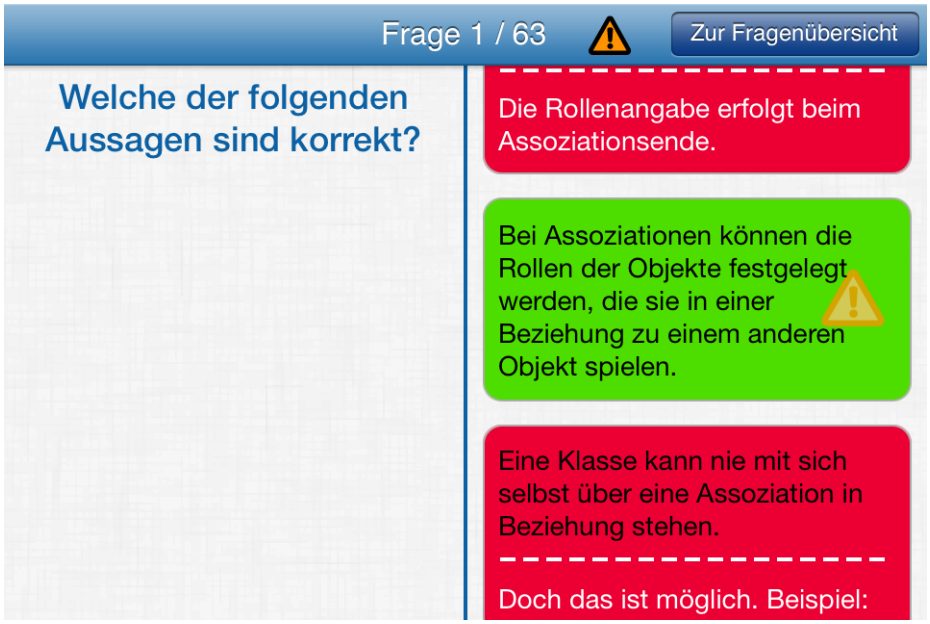

Abbildung 6. Feedback-Ansicht der iOS - Bestandsapplikation

Wenn man ein Quiz beendet, bekommt man eine Übersicht über die richtigen und falschen Antworten. Eine Antwortmöglichkeit wird dabei dann mit einem Häkchen oder Rufzeichen symbolisiert, sowie den Farben Rot und Grün.

Die Nutzergruppe konnte nicht feststellen, welche ihrer Antwortmöglichkeiten, die sie selbst ausgewählt hatten, richtig ist bzw. ob die grün hinterlegten Antworten bedeuten, dass ihre gewählte Antwort richtig ist oder, das die richtige Antwort gewesen wäre. Außerdem stehen bei grün hinterlegten Antworten Rufzeichen dabei, sodass niemand genau wusste, was das letztliche Resultat überhaupt bedeutet.

Unterhalb falscher Antworten gab es vereinzelt eine strichlierte Linie, die eine Art Zusatzinformation ausgibt. Ein Teil der NutzerInnen meinte, dass die rechte Hälfte mit den Antwortmöglichkeiten dadurch eine lange Schlange bildet und unübersichtlich wird. Es wurde vorgeschlagen das Symbolmanagement zu verbessern und ein Symbol einzuführen, welches anzeigt, dass Feedback vorhanden ist. Wenn dies der Fall ist, soll man mit einem Klick darauf diese Information einblenden, sowie ausblenden können, um sich nur die Zusatzinformationen abholen zu können, die auch die Nutzerin/den Nutzer interessieren um diese/diesen nicht sofort mit Text zu überschütten.

Aufgrunddessen, dass unsere Nutzergruppe aus den verschiedensten Nutzern

Abbildung 7. Lernkartei-Ansicht der iOS - Bestandsapplikation

bestand, fand sich zufällig ein Nutzer mit einer Rot-Grün Sehschwäche. Dieser konnte keinen wesentlichen Unterschied zwischen den roten und grünen Antworten feststellen. Nur das Rufzeichen war für ihn ersichtlich und deutete für ihn daraufhin, dass die Antwort falsch ist. Ein Lösungsvorschlag wäre die Möglichkeit Alternativfarben wählen zu können.

Zuletzt wurde noch angemerkt, dass die eingeblendete Aktivität, dass wenn eine Frage richtig beantwortet wird, diese in die nächste Lernkartei verschoben wird, als verwirrend angesehen wird. Diese Seite wird aufgerufen zwischen "Quiz beenden" und der Möglichkeit die Korrektur anzusehen. Wenn eine Frage richtig ist, dann soll sie in die nächste Lernkartei wandern, jedoch muss das nicht so präsent dargestellt werden.

Es wurde eine Befragung durchgeführt, ob das Lernkarteisystem weiterhin im System enthalten sein soll. Die NutzerInnen haben sich einstimmig darauf geeinigt, dass das System den Lernfortschritt maßgeblich fördert und sinnvoll ist. Dieses System ist in Österreich bei jungen Erwachsenen bekannt, denn die theoretische Führerscheinprüfung basiert auf diesem Prinzip. Eine Benutzerin hat angemerkt, dass bei einer Alternativapplikation dieses Feature fehlt. Es ist wichtig, Feedback darüber zu erhalten, wie viele Fragen noch nie richtig beantwortet wurden. Das "Prinzip des Versuchs und Irrtum" nach Edward Lee Thorndike stützt diese Art der Unterstützung beim Lernprozess, da die Nutzerin/der Nut-

zer so lange die Übungen durchführen kann, bis sie vollständig richtig gelöst wurden. Die Motivation zum Lernen wird damit erreicht, indem das Bedürfnis besteht, alle Fragen in die beste Box ablegen zu können. Aus theoretischer Sicht ist es vorteilhaft, die Fragen nach einer bestimmten Zeit, sofern sie nicht beantwortet wurden, wieder eine Box zurück zu stufen. Damit soll die Lernfrequenz gesteigert werden, um den Lernerfolg dauerhaft sicherstellen zu können [15].

Die NutzerInnen wurden ebenfalls befragt, ob sie sich ein anderes System wünschen würden, welches den Lernfortschritt steigern könnte, jedoch ist keiner Nutzerin/keinem Nutzer ein konträres System verglichen mit dem Lernkartensystem eingefallen oder bekannt gewesen.

3.6 Analyse theoretischer Designansätze

Einhergehend mit der Analyse theoretischer Designprinzipien wurden folgende Ansätze erarbeitet, die schließlich mit den Anwendern beim partizipativen Designprozess untersucht und beurteilt wurden.

Für die Überarbeitung des Layouts bieten sich mehrere Möglichkeiten an. Am Beispiel der derzeitig verwendeten Darstellung (siehe Abbildung 8), werden folgende Änderungsvorschläge nach einhergehender Literatursuche in Erwägung gezogen. Im Anschluss an die Recherche wurden Skizzen angefertigt, die für die Diskussion mit möglichen Anwendern herangezogen wurden.

Der Fragetext sollte von zentriert hin zum Blocksatz umformatiert werden, der bei technischen Dokumentationen auch häufig zum Einsatz kommt. Damit wird das Lesen durch eine allgemein ruhigere Wahrnehmung, ohne dauernd den Zeilenanfang suchen zu müssen, unterstützt [14]. Ein schlecht aufgeteilter Blocksatz kann sich aber ebenso negativ auswirken, Linksbündiger Text wäre demnach dann vorzuziehen, sollte sich die Blockformatierung bei Usability-Tests als ungeeignet herausstellen [9]. In der Praxis hat sich gezeigt, dass die vordefinierten Steuerelemente von Android einen Blocksatz nicht anbieten.

Eine vertikale Aufteilung des Textes über die gesamte Breite des Bildschirms ist am Mobilgerät ebenfalls besser für den Lesefluss. Es wird ein Zeichenumfang von 30-40 Zeichen je Zeile empfohlen [14]. Dies entspricht etwa der doppelten Breite an Platzverbrauch für den Fragetext wie bisher und nimmt dann die gesamte Breite des Bildschirms ein. Die Lesbarkeit hat sich laut Aussage einiger Testnutzer verbessert, da sich die Anzahl der Zeilen im Vergleich zu den Abbildungen der iOS Anwendung sichtlich verringert haben.

Bei der Darstellung der richtig und falsch beantworteten Fragen ist es vorstellbar, unterschiedliche Schriftarten oder Stile einzusetzen. Erschwerend kommt hier hinzu, dass eine Darstellung auf verschiedenen Geräten möglicherweise nicht überall in gleichem Maße von Vorteil ist. Für das UML-Quiz wurde ausschließlich auf die Textgröße geachtet und die Schriftart, Farbe und Stil in der Anwendung

Abbildung 8. Farbgestaltung der iOS - Bestandsapplikation

zur Gänze gleich gehalten.

Das "Gesetz der Nähe" [14] könnte bei der Anordnung der richtig und falsch beantworteten Fragen zum Tragen kommen. Richtig beantwortete Fragen werden als erste Elemente im Layout angeordnet, mit dem Ziel, dass sich diese das Gehirn am besten einprägen wird. Es folgen schließlich nicht gekreuzte, mit etwas Abstand versetzt falsch beantwortete Fragen. Dem Unbewusstsein wird so die Nichtzugehörigkeit zur Menge richtiger Antworten vermittelt. Falsche Antworten, die auch nicht vom Anwender ausgewählt wurden, sollten gar nicht angezeigt werden. Dann würde man aber nicht mehr, im Gegensatz zur bisherigen Anwendung, Erklärungen zu jeder Antwortmöglichkeit abfragen können.

Das "Gesetz der Gleichartigkeit" wird dabei auch genutzt, um den Eindruck auf das Unbewusstsein zu verstärken. Diese psychologischen Denkansätze wurden letztendlich verworfen, da die Testpersonen davon ausgingen, dass diese die Anwenderin/den Anwender in Folge eher verwirren würde, als beim Lernprozess behilflich zu sein.

Um das Problem der Wahrnehmung für rotblinde (Protanopia) Personen zu verdeutlichen, wurde ein Screenshot der bisherigen Anwendung mit einem Filter behandelt, der ungefähr dem Sehvermögen einer betroffenen Person nahe kommt [24]. Der Einsatz von roten und grünen Farbtönen ist, wie bisher um-

Abbildung 9. Protanopia-Filter auf Feedback-Ansicht

gesetzt, nicht geeignet, um richtige und falsche Antworten voneinander zu trennen. Eine Unterscheidung ist für Personen mit Protanopia nicht möglich. Um eine Bedienbarkeit unabhängig von der Fehlsichtigkeit zu gewährleisten, wurde im Designprozess eine Einstellungsoption erarbeitet. Farbe kann, bewusst eingesetzt, brauchbare Information für die Nutzerin/den Nutzer (siehe Kapitel 4.3.5) enthalten, die aufgrund der gesteigerten Aufmerksamkeit besser wahrgenommen wird [9].

Um den Usability-Anforderungen von Android Applikationen gerecht zu werden, sind geringe Anpassungen an der GUI notwendig. Das UML-Quiz für Android wird durchgängig eine Action Bar erhalten. In dieser können die Hilfe und die Einstellungen aufgerufen werden, sowie innerhalb der Anwendung navigiert werden, unabhängig davon in welchem Zustand sich diese befindet. Das Einstellungsmenü wird vollständig in die Action Bar integriert.

3.7 Analyse von e-Learning Quiz Applikationen

Im letzten Analyseschritt werden Alternativapplikationen und deren Interaktionsmechanismen untersucht. Das Hauptaugenmerk wird auf die Darstellung einer Frage und der zugehörigen Antworten gelegt. Die Frage die es sich zu stellen gibt lautet: "Wie haben andere das Problem der Darstellung von Fragen gelöst?"

Die erste zu untersuchende Applikation ist "Anki" (für iOS) und "AnkiDroid"

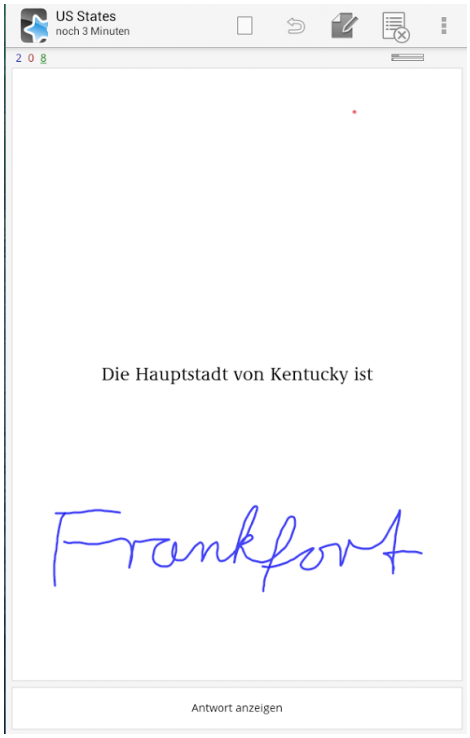

Abbildung 10. Ankidroid für Android [4]

(für Android). Diese Applikation wurde uns von einer Nutzerin vorgeschlagen. Anki ist eine sehr vielfältige Lernkarteiapplikation und basiert auf dem Prinzip, dass es zu jeder Frage nur eine Antwort gibt. Bei der Analyse des Designs wurde die freundliche Helligkeit positiv bewertet. Ein weiterer sehr guter Aspekt sind die Menüpunkte am oberen Bildschirmrand. Dies ermöglicht es, mehr Platz für den Inhaltsbereich zur Verfügung gestellt zu bekommen und trotzdem Funktionen zur Verfügung zu stellen.

Die nächste Applikation wurde uns ebenfalls von einer Nutzerin vorgeschlagen und heißt "BRAINYOO". Brainyoo besitzt einen Großteil der Elemente, die auch in der UML-Quiz Applikation verwendet werden. Eine Frage kann aus einem Text und / oder einem Bild bestehen. Die Frage wird zwingend im Hochformat dargestellt und bietet somit einen vertikalen Modus. Unter der Frage werden die Antwortmöglichkeiten dargestellt. Die meisten NutzerInnen haben sich sofort für dieses Layout festgelegt. Es ist klar strukturiert, jedoch könnte die Frage weniger Platz einnehmen. Durch vertikales Scrollen gelangt man zu den restlichen Antwortmöglichkeiten und schiebt den Fragetitel nach oben aus dem Bildschirm hinaus. Durch einen Klick auf das Bild in der Frage wird dieses vergrößert.

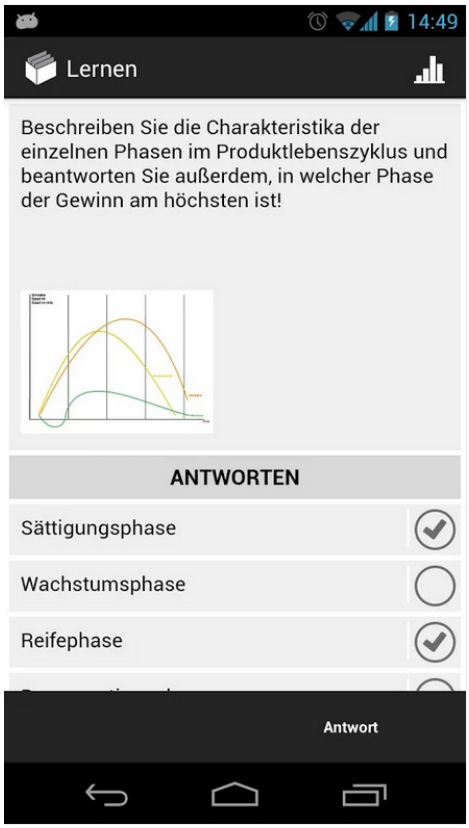

Abbildung 11. BrainYoo für Android [5]

"CoboCards" ist ebenfalls eine auf Karteikarten basierende Applikation. Diese Applikation war vorher noch keiner Nutzerin/keinem Nutzer bekannt. Der Balken am oberen Bildschirmrand wurde positiv angemerkt. Er dient dazu anzuzeigen, wie viele Fragen noch zu lösen sind. Die Abhebung von Frage und Antwort mittels Strich wurde ebenfalls positiv angemerkt.

Als letzte Applikation wurde der Nutzergruppe "Repetico" vorgestellt. Das Besondere an dieser Applikation war die Einfachheit des Designs und das weiche Auftreten. Den Nutzern hat diese Vereinheitlichung des Designs besonders gut gefallen.

Außerdem wurden noch sechs weitere Applikationen vorgestellt, die jedoch nicht maßgeblich zum entscheidenden Design beigetragen haben. In dieser Analysephase stellte sich heraus, dass das Interaktionskonzept von Brainyoo am besten bei den Nutzern angekommen ist. Die klar strukturierte Aufteilung zwischen

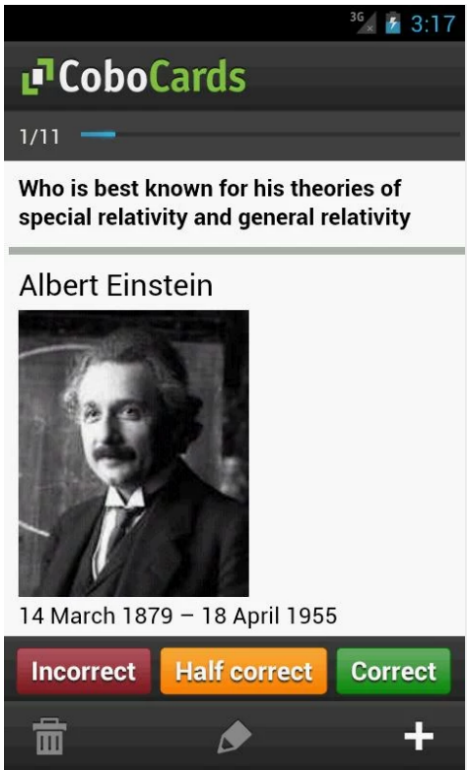

Abbildung 12. CoboCards für Android [6]

Fragen und Antworten, sowie das mögliche Einbinden von Bildern, wie es für die zu erstellende e-Learning Applikation wichtig ist, haben die NutzerInnen überzeugt.

4 Praxis

In der Praxis geht es um die Umsetzung der ausgearbeitet Anforderungen. Die Analyseerkenntnisse werden in Entwürfen ausgearbeitet und den Nutzern erneut zur Vorlage präsentiert. Anschließend steigt man in den iterativen Entwicklungsprozess ein und beginnt mit der Entwicklung der Applikation unter Berücksichtigung der ausgearbeiteten Detailgrade mit den Nutzern.

4.1 Analyseerkenntnisse

Die Forderungen der Nutzergruppe waren teilweise eindeutig. Einer der Hauptstörfaktoren am Interaktionsdesign der iOS - Bestandsapplikation liegt in der Ausrichtung der Fragen. Es wurde der Wunsch geäußert eine vertikale Ansicht

Abbildung 13. Repetico für Android [21]

der Fragen zu erstellen, weil die Ansicht natürlicher für den Menschen ist und auch erwartet wird. Das Scrollverhalten soll hierbei auf vertikales Scrollen begrenzt werden, sodass bei längeren Fragen und Antworten man einfachst nach oben oder unten wandern kann. Zusätzlich wurde von einer Nutzerin angemerkt, dass das Quiz-Layout eine Anlehnung an das Layout der eigentlichen Prüfung haben soll und das wiederum impliziert, dass bei einer Prüfung die Frage über den Antworten steht.

Die Ausrichtung der iOS - Bestandsapplikation war nur auf den horizontalen Modus beschränkt, was dazu führt, dass man automatisch beide Hände zum Benutzen benötigt. Dies soll geändert werden, sodass auch ein vertikalen Modus inkludiert ist. Das bringt den Vorteil, dass in Situationen, wo man nur eine Hand zur Verfügung hat, ein Quiz absolviert werden kann, welches durch das Scrollverhalten der Applikation unterstützt wird.

Das Design der Feedback-Ansicht wird grundlegend geändert. Wenn eine Antwort richtig beantwortet wurde, so wird diese mit einem Häkchen versehen. Eine Antwort, die nicht beantwortet wurde, obwohl sie richtig gewesen wäre, bekommt ein Rufzeichen und eine Antwort, die fälschlicherweise als richtig markiert wurde, bekommt ein Kreuz. Zusätzlich werden richtige Antworten mit einem dickeren Rahmen versehen, um den Effekt der Transparenz zu erhöhen. Außerdem wird das Farbkonzept angepasst, sodass richtig beantwortete Fragen am linken Rand grün und falsch beantwortete Fragen rot hinterlegt werden. Dieses Farbkonzept kann jedoch verändert werden, sodass auch Menschen mit einer Farbblindheit bevorzugt werden können.

Das Anzeigen des Feedbacks zu einer Antwort soll nicht mehr automatisch bei allen Antwortmöglichkeiten geöffnet sein, sondern mit einem Klick darauf geöffnet oder geschlossen werden können. Somit können sich die NutzerInnen genau die Informationen genauer ansehen, die sie auch interessieren und werden nicht mit Informationen am mobilen Endgerät überhäuft. Falls es ein Feedback zu einer Antwort gibt, wird dieses durch ein Fragezeichen am linken Rand der Antwort symbolisiert.

Das Lernkarteisystem wird vollständig übernommen, denn viele NutzerInnen kennen die Vorteile dieses Systems von der österreichischen Führerscheinprüfung und konnten ihren Lernerfolg somit besser steuern, denn wenn ein Großteil der Fragen in der dritten Box waren, dann wussten sie, dass sie den Stoff gut verstanden hatten.

Das Layout der Fragen wird sich grundlegend am Layout von Brainyoo orientieren. Die Darstellung der Frage mit der Einbindung eines Bildes, sowie der Darstellung der Antworten hat den Nutzern am ehesten und sinnvollsten zugesprochen und unterstreicht nochmals das Layout einer Frage im vertikalen Design. Zusätzlich soll mit einer Berührung des Bildes, dieses im Vollbildmodus angezeigt werden, um alle Detailgrade gut erkennen zu können.

Das Design soll sich an die klare Struktur von Repetico anpassen. Durch eine farbliche Abhebung der Frage vom Antwortteil soll ein Unterschied merklich herbeigeführt werden. Der Antwortteil soll einfach und freundlich gestaltet werden. Durch die Berührung einer Frage soll diese eine verstärkte Umrandung erhalten, um eine Selektion deutlich zu erkennen.

Aufgrunddessen, dass es unter Android die Möglichkeit von Menüs gibt, wird ein Menü in allen Seiten der Applikation eingebaut. Auf der Hauptseite kann man bestimmte Einstellungen darin definieren, sowie zu den Zusatzinformationen gelangen. Innerhalb einer Frage bietet es die Möglichkeit sich Informationen über den Modus vom Quiz zu holen oder die Darstellung der Farben im Feedback zu ändern, wenn gewünscht.

Die Erkenntnisse dieser Analyse werden nun in einen Entwurf (Mockup) gepackt und den Nutzern vorgestellt.

4.2 Entwurf

Der Entwurf (Mockup) gestaltet sich aus den Ergebnissen der Analysephase mit den Nutzern. Er widerspiegelt die gewonnen Erkenntnisse und zwar so wie es ein Dritter herauslesen würde. Es wurden drei Mockups erzeugt. Eines für den vertikalen Modus und das andere für den horizontalen Modus der Applikation und das Dritte für den Feedback-Modus der Frage.

Abbildung 14. Entwurf der Fragenansicht im vertikalen Modus

Abbildung 15. Entwurf der Fragenansicht im horizontalen Modus

Der Nutzergruppe wurden die Mockups vorgelegt. Sie konnten sich grob vorstellen wie es aussehen soll und fühlten sich bestätigt. Zwar war die Darstellung mithilfe der Mockups nicht am genauesten, jedoch konnten sie erkennen, was das Ziel ist. Nachdem die Entwürfe einstimmig akzeptiert wurden, kann mit der Entwicklung der Applikation am Betriebssystem Android begonnen werden.

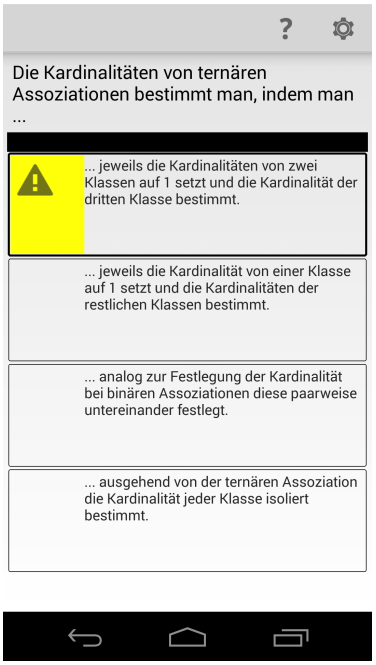

Abbildung 16. Entwurf der Feedback-Ansicht

4.3 Iterativer Entwicklungsprozess

Der Entwicklungsprozess wird in Iterationsschritten durchgeführt. Im Rahmen dieser Bachelorarbeit wird die erste Iteration durchgeführt. Das bedeutet, dass das System aus den vorhandenen Erkenntnissen und Mockups entwickelt wird. Das Ergebnis der ersten Iteration wird der Nutzergruppe vorgelegt. Diese führt anschließend Tests durch, bewertet und kritisiert diese. Das Ergebnis dieser Evaluierungsphase wird anschließend in die nächste Iteration gepackt und dort ausgebessert bis ein Systemzustand erreicht worden ist, der die Nutzergruppe zufrieden stellt.

Die Nutzerrückmeldungen sind einmal während einer geplanten Fokusgruppe in das Design eingeflossen, mehrmals bei den zweiwöchentlichen Treffen des Bachelorseminars, aber insbesondere zwischendurch indem StudienkollegInnen aller Fachrichtungen Designprototypen beurteilt und Verbesserungsvorschläge mitgeteilt haben. Viel wichtiger war es, möglichst viele verschiedene Arten von Anwendern zu befragen, als große Treffen zu veranstalten. Einerseits aus Zeit- und Organisationsgründen, andererseits weil die wichtigsten Fragen bereits für die Fokusgruppe ausgearbeitet und diskutiert wurden. Die Meinung in der Fokusgruppe war großteils einstimmig zu den Kritikpunkten und es konnten einige Designempfehlungen, auf denen spätere Entscheidungen basierten, ausgearbeitet werden.

Für die zweite Evaluation wurde mit wenigen Nutzern gemeinsam die Android Anwendung ausgetestet und nach Kritik gefragt. Hauptsächlich aber wurde die Installationsdatei per Mail verteilt und die NutzerInnen angewiesen, diese zu installieren und bezüglich Bedienung sowie Verständlichkeit zu prüfen.

4.3.1 Darstellung der Fragen Die Erkenntnisse aus der Anforderungsanalyse, sowie der Nutzerevaluierung wurden für die Erstellung neuer Steuerelemente herangezogen. Die Abbildung 17 zeigt einen ersten Entwurf und vergleicht diesen mit der iOS-Bestandsapplikation in Abbildung 5.

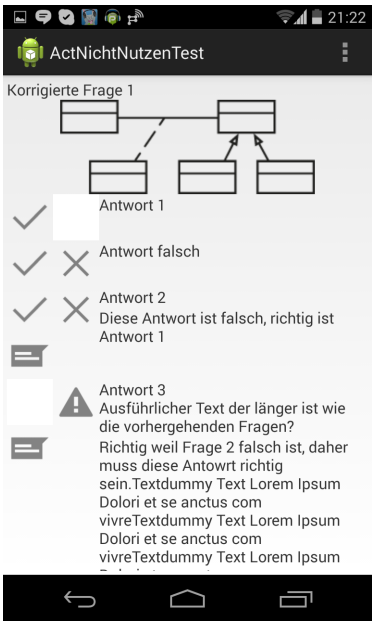

Abbildung 17. Entwurf der Feedback-Ansicht

Konträr zur iOS Anwendung, ist erkennbar, dass Navigationssymbole bei der Android Portierung bewusst weg gelassen wurden. Die Speichern- und Beenden-Funktion erwartet die Benutzerin/der Benutzer einer Android Anwendung als gegeben. Die Beenden-Funktion wird mittels BackStack und BackButton gelöst. Das Layout wurde entsprechend den Mockups verändert. Als Icons kommen die offiziell zur Nutzung von Android frei gestellten ActionBar-Icons zum Einsatz.

In der Abbildung 17 ist eine aufgelöste Frage vorzufinden. Dabei war festzustellen, dass nicht nur eine Trennung anhand eines Abstandes zwischen den Steuerelementen, sondern durch eindeutige gezeichnete oder farbliche Hervorhebung geboten werden muss. Die Icons auf der linken Seite bedeuteten je Antwort: wurde angekreuzt (li. Oben, wenn nicht gekreuzt bleibt dieser Bereich leer), falsch gekreuzt (re. Oben, wenn richtig beantwortet bleibt dieser Bereich leer) ist die Antwort falsch dann wird ein "X" angezeigt, ist sie richtig wurde aber nicht angekreuzt ein "!", es ist eine Erklärung verfügbar (li. Unten). Es war offensichtlich, dass die Oberfläche dadurch überladen wirkt, somit wurde entschieden nur ein Symbol darzustellen und optional je nach Verfügbarkeit der Erklärung ein weiteres.

Abbildung 18. Entwurf der Fragen-Ansicht

In Abbildung 18 ist die Benutzeroberfläche nach der Umgestaltung ersichtlich. Das obere Icon ist entweder ein Hakerl (Antwort richtig beantwortet), Kreuz (Falsch beantwortet), Rufzeichen (Antwort ist richtig und wurde nicht beantwortet). Das untere Icon (steht Sinnbildlich für textuelle Nachrichten) wird angezeigt bei vorhandenen Erklärungen. Es wurde von befragten Nutzern nicht als solches erkannt. In einer kurzen Diskussion mit StudienkollegInnen einigte man sich auf das "?", da dies als ein Synonym für Hilfe, Unklarheit allgemein an-

gesehen wird. Während des Bachelorseminars wurde von mehreren anwesenden StudentInnen vorgeschlagen das "?" auf die rechte Seite der Antwort zu verlegen. Der Kontext, ob die Frage richtig oder falsch ist, wäre demnach immer auf der linken Seite zu finden. Der andere wäre (ob zusätzliche Information existiert) nur dann existent, wenn auf der rechten Seite zur Verfügung gestellt. Ein weiterer Vorschlag war, die Bedeutung der Symbole ebenfalls, egal wie gut diese bekannt sein sollten, in einer Hilfe zu erklären.

Der frühe Stand der Benutzeroberfläche wurde nebenbei mehreren Kolleginnen und Kollegen vorgeführt. Es stellte sich heraus, dass ein weiteres Manko der iOS Anwendung in der Android Portierung Einkehr gefunden hatte. Für AnwenderInnen, deren die Funktionalität unbekannt war, war es nicht selbstverständlich, dass die Antworten zu markieren sind. Android bietet hierfür ein eigenes Steuerelement (eine Checkbox wie von vielen Benutzeroberflächen bekannt), um dieses Bedienungsparadigma beizubehalten war es notwendig eine leere Checkbox bei jeder Antwort anzuzeigen. Es stellte sich in weiterer Hinsicht die Frage, ob die von Android mitgelieferte oder eine eigene Implementierung mit dem großen Icon zu verwenden ist. Im Sinne eines gleichartigen Designs wurde entschieden, eine entsprechendes Steuerelement selbst zu programmieren.

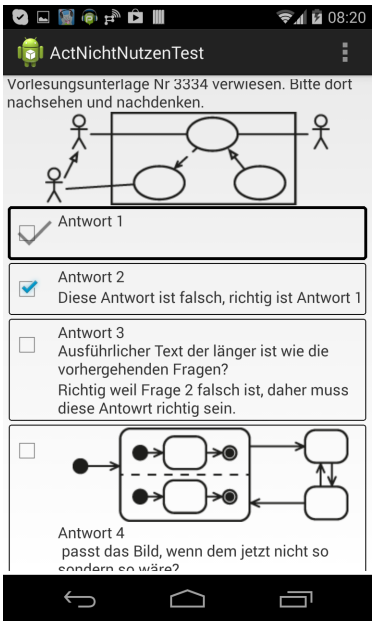

Abbildung 19. Entwurf der Fragen-Ansicht

Auch war bei den ersten Implementierungen für die TestnutzerInnen trotz Hakerl nicht eindeutig ersichtlich, ob die Frage beantwortet war (siehe Abbildung 19). Für diese war es jedoch wichtig, dass bereits beantwortete Antworten besser hervorgehoben werden sollten. Somit wurde der Rahmen bei beantworteten Elementen verstärkt hervorgehoben.

4.3.2 Farbliche Gestaltung An der iOS Applikation wurde die farbliche Gestaltung besonders kritisiert, auch in Anbetracht der erschwerten Erkennbarkeit für Personen mit Farbschwäche [24]. Im nächsten Schritt wurde daher die Farbgebung eingearbeitet. Da von einer übermäßigen Nutzung von Farbe abzusehen ist, wurde die farbliche Hinterlegung auf den Bereich der Symbole beschränkt (siehe Abbildung 20).

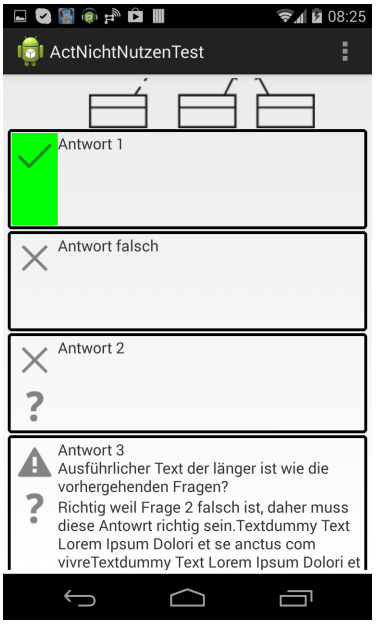

Abbildung 20. Entwurf der Fragen-Ansicht

Letzten Endes wurde gewünscht, dass die Antwort zur Gänze mit Farbe betont wird. Als Kompromiss wird eine zusätzliche Einstellungsoption angeboten. Für die Farbgestaltung gibt es eine Einstellungsmöglichkeit. So kann ein Farbschema (Rot-Grün, Blau-Gelb, Grau-Grau, Keine Farbe) vom Benutzer individuell

ausgewählt werden. Die Farbschemata sind dem Umstand geschuldet, dass Farbfehlsichtigkeit oder gar das Fehlen von Farbe den Betroffenen eine Bedienung erschweren würde, sofern ein unpassendes statisches in das Programm eingebaut werden würde. Für die Darstellung von Bildern wäre eine Transformation der Farbe angebracht [24].

4.3.3 Grafische Trennung zwischen Fragetext und Antwort Eine grafische Abtrennung der Antworten von den Fragen wurde von den Testanwendern gewünscht. Die iOS Anwendung trennt diese mit einem Strich voneinander, andere in der Analyse gefundene Anwendungen setzen auch auf farbliche Gestaltung. In der Abbildung 21 ist die misslungene farbliche Umsetzung erkennbar. Das Design wirkte bereits zu bunt, insbesonders wenn bei der Lösung noch die Fragen zum Teil oder gänzlich farblich hinterlegt werden. In der Abbildung21 ist auch bereits das Menü für die Hilfestellung und Einstellungen sichtbar.

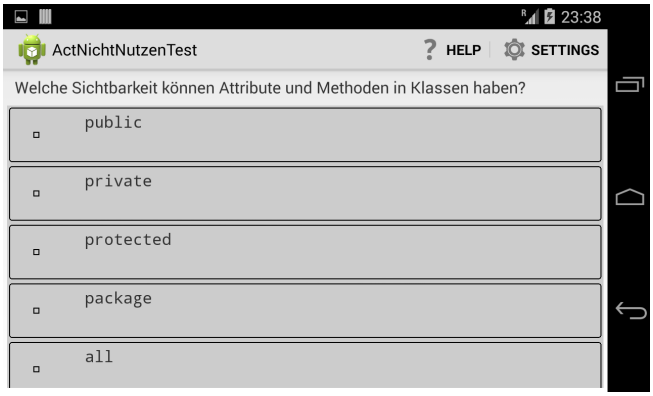

Abbildung 21. Entwurf der Fragen-Ansicht

Die farbliche Trennung wurde schließlich nicht weiter verfolgt. Es wird die Frage von den Antworten mit einem schwarzen Balken getrennt (siehe Abbildung 22).

4.3.4 Hilfestellungen für die NutzerInnen Da der Einsatz von Icons besondere Bedachtnahme erfordert, betreffend korrekter Interpretation durch die Anwenderin/den Anwender, wurde festgestellt, dass eine Erklärung in Form eines Tutorials oder abrufbar in der Hilfe unumgänglich erscheint. "Der ISO/IEC CD 11581-1.2 Entwurf aus dem Jahre 1993 empfiehlt, dass sofern zwei Drittel der

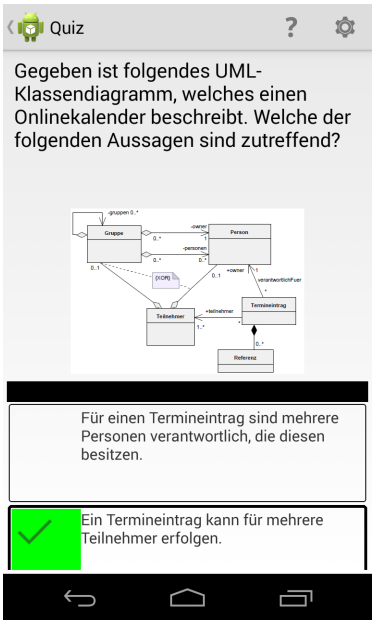

Abbildung 22. Entwurf der Fragen-Ansicht

AnwenderInnen ein Icon verstehen, dieses eingesetzt werden kann. Jedoch stellt sich die Frage, ob ein Drittel der AnwenderInnen außer Acht gelassen werden darf" [20]. Die unterschiedliche kulturelle Herkunft der NutzerInnen erschwert ebenfalls die Wahl des geeigneten Icon.

Die Abbildung 23 zeigt den ersten Entwurf für das Einstellungsmenü, mit dem die Nutzerin ein Farbschema für die korrigierten Antworten auswählen kann.

Von den Nutzern wurde bemängelt, dass die Funktionalität nicht klar ersichtlich war. Auch hat das Ändern des Farbschemas keine sofort ersichtliche Reaktion und widerspricht damit dem Usability Gedanken, dass bei Interaktion eine Rückmeldung vom System an die Anwenderin/den Anwender erfolgt. Daher wurden Beispielantworten der Benutzeroberfläche beigefügt, deren Farbe sich sofort bei geänderter Auswahl der Optionen anpasst (siehe Abbildung 24).

Einige Android Anwendungen zeigen bei der erstmaligen Verwendung Hilfsdialoge an, die über Gesten und Steuerung die Benutzerin/den Benutzer informieren soll. Die iOS Anwendung zeigt bei jedem Start eines Quiz der Benutzerin an,

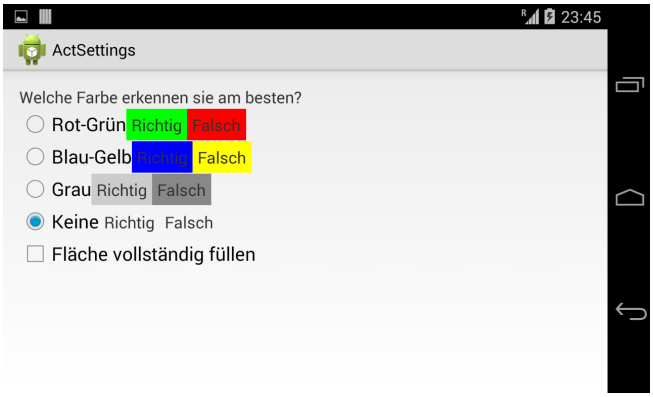

Abbildung 23. Entwurf des Einstellungsmenüs

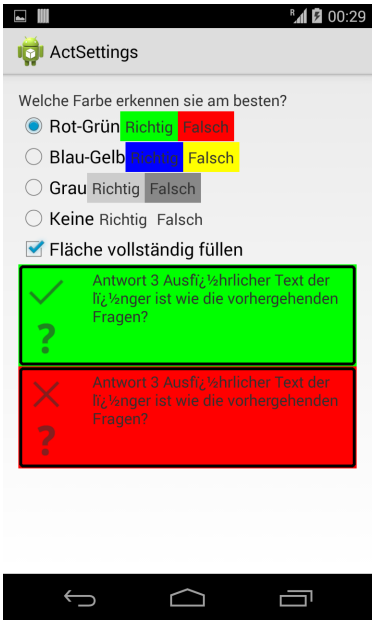

Abbildung 24. Entwurf des Einstellungsmenüs

dass diese nach rechts swipen muss. Dies ist für die Beginnerin/den Beginner hilfreich, für fortgeschrittene NutzerInnen könnte diese Mitteilung allmählich als lästig empfunden werden, da sie bereits Kenntnis über diese grundsätzlich hilfreiche Information besitzen. Es bietet sich an diese Hilfestellung nur die ersten Male anzuzeigen. Darüber hinaus muss die Benutzerin die Möglichkeit haben, jederzeit die Hilfe aufrufen zu können. Im Unterschied zur iOS Anwendung, zeigt die Android Portierung den BenutzerInnen mehr Informationen an (siehe Abbil-

dung 25).

Abbildung 25. Entwurf der Hilfe

So wird beim Starten eines Quiz die Steuerung zur Gänze erklärt. Wenn alle Fragen beantwortet wurden, wird die Benutzerin aufmerksam gemacht, dass sie zu den Fragen zurückkehren kann oder die Lösung angezeigt bekommt. Gleichzeitig wird auch die Bedeutung der verwendeten Icons bei der Auflösung erklärt (siehe Abbildung 26).

Für die Erklärung der Steuerung wurden die Bezeichnungen dem "Touch Gesture Reference Guide" entnommen [17].

Gelangt die Benutzerin die ersten Male zu den korrigierten Fragen, so wird der Einstellungsdialog für das Farbschema angezeigt. Ziel ist es, mit dieser Strategie den Benutzer mit wenig Information im passenden Moment zu versorgen, um eine Überforderung zu verhindern. Die BenutzerInnen werden bewusst von Einstellungsmenüs mit vielen unnützen Optionen fern gehalten.

Sind alle gelösten Fragen vom Benutzer eingesehen worden, so wird schließlich nur mehr der Text "Quiz beenden" und ein Symbol, das die Swipe-Interaktion

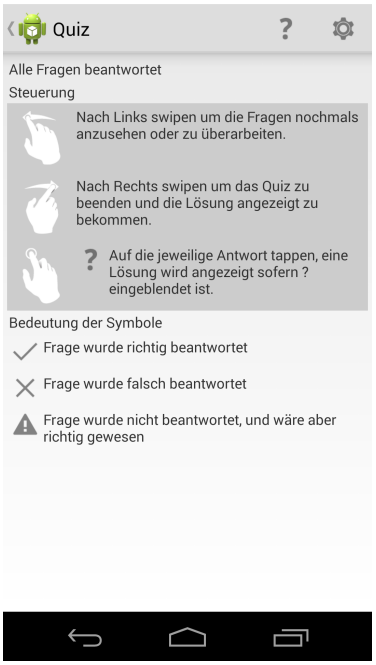

Abbildung 26. Entwurf der Hilfe

andeutet, angezeigt. Vorhergehende Entwürfe mit mehr Inhalt wurden verworfen, da dem Benutzer mit wenigen Worten ohnedies klar erscheint, was die folgende Interaktion bewirken wird (siehe Abbildung 26 und 27).

4.3.5 Accessibility Für die Nutzung des Text-to-Speech (auch Talk-Back genannt) Features von Android ist es notwendig die Eigenschaft "Content Description" mit einem Steuerelement des beschreibenden Textes zu versehen. Aktiviert die Nutzerin/der Nutzer die Funktion, wird der beschreibende Text vorgelesen, sobald mit dem Finger ein Steuerelement berührt wird. Die entsprechende Eigenschaft wurde bei der Erstellung des Programms in den dynamisch generierten Steuerelementen (Quiz-Frage, Antwortmöglichkeiten) gesetzt. Standardmäßig wird der angezeigte Text vorgelesen oder der Bezeichner des Steuerelements vorgelesen.

Das Talk-Back Feature wurde auf einem Android Smartphone mit Version 4.4.3 mit einem Testbenutzer ausgetestet. Dabei wurde eine Version des UML-Quiz verwendet, deren Entwicklungsstand sowohl in Funktion als auch Design fast fertig implementiert war. Im Vorfeld machte die normalsichtige Person sich mit der Bedienung bei eingeschaltetem Talk-Back vertraut. Schließlich wurden der

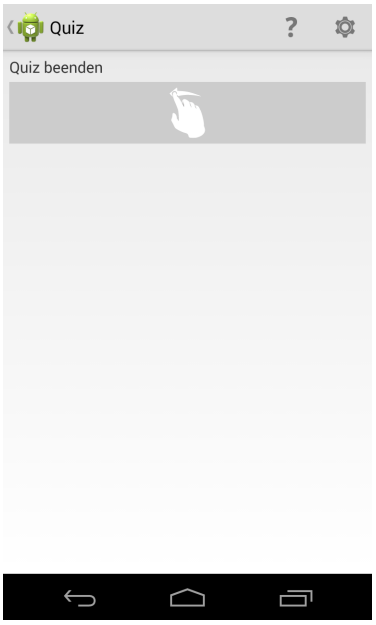

Abbildung 27. Entwurf der Quiz beenden Ansicht

Testperson die Augen verbunden, da das Feature für die Testperson gänzlich unbekannt war, half ein Entwickler stets bei der Bedienung mit.

Ein Problemfall war bereits das Startmenü, für das zwar Beschreibungen für die Textboxen existierten, aber die umgebenden Miniaturbilder keine hatten. Der Benutzer musste genau den klein geschriebenen Text treffen. Für das Layout ist entscheidender, dass die einwandfreie Navigation für die Touch-Bedienung gewährleistet ist als die für sehende Personen optimale Darstellung, bei der auch die Textlänge berücksichtigt werden muss.

"Fitts Law besagt: Je näher größer ein Objekt ist, umso schneller kann es bei der Eingabe erreicht werden" [9]. Man kommt in Folge wohl nicht daran vorbei, zwei optimierte Benutzerschnittstellen zu erstellen, nämlich eine für Sehende und eine für Blinde und Personen mit stark eingeschränkter Sicht. Es wurde im weiteren Verlauf ersichtlich, dass sämtliche Symbole und farbliche Elemente mit in die Beschreibung einfließen mussten. Ob Fragen beantwortet oder nicht beantwortet wurden, war ebenfalls vergessen worden. Es wurden lediglich standardisierte Ansagen von Android über die Steuerelemente ("Zum aktivieren betätigen") wiedergegeben. Wichtig ist auch, dass Elemente immer an der selben Stelle zu finden sind. Die Idee von einem Hauptelement, das auf weitere Elemente aufmerksam macht, entstand, als der Beobachter feststellte, dass die Testperson bei erklärenden Folien vielfach Steuerelemente nicht entdeckte. Dieses Haupt-

element sollte immer als erstes angesteuert werden oder gar beim erstmaligen Auftauchen der Benutzeroberfläche vorgelesen werden.

Kritisiert wurde, dass keine Rückmeldung über den Start der Anwendung ausgegeben wurde. Das Hauptmenü wurde bereits viele Sekunden angezeigt, bevor auf anraten des Beobachters der Testnutzer die Interaktion begann. Eine Art "Willkommenstext" würde in diesem Fall Abhilfe schaffen. Die Testperson erwähnte auch, dass es wichtiger wäre, ein Feedback über den derzeitigen Zustand der Anwendung zu erhalten. Darunter verstand die Testperson, dass zum Beispiel "Frage 3 von 10", "Lösung 1 von 10", "Diese Frage hat 7 Antwortmöglichkeiten" vorgelesen wird.

Auch für Auswahldialoge, im Speziellen die Auswahl, ob ein bereits gestartetes Quiz fortgesetzt werden soll, ließen den Testbenutzer im Unklaren welche Eingabemöglichkeiten bestehen. Während der weiteren Bedienung fiel auf, dass vom Android Betriebssystem aufgehende Dialoge dieses Feature aber implementiert hatten. Dies ist ein konkreter Grund, warum bei der Softwareentwicklung das Rad nicht neu zu erfinden ist, sondern vom System zur Verfügung gestellte Interaktionsmechanismen mit Vernunft Gebrauch gemacht werden sollte. Dies bekräftigt auch die Vorschläge der Softwareentwickler, die Activities der Anwendung zu reduzieren (siehe Kapitel 6.1.1).

Obwohl der Test weder vorbereitet oder mit einer Person, die über eingeschränkte Sicht verfügt ausgeführt wurde, konnten dennoch wichtige Designprinzipien erarbeitet werden. Eine weitere Analyse oder Implementierung des Features blieb aus, da ein Großteil der Fragen über Grafiken und Bilder verfügen, die nicht automatisiert vorgelesen werden können. Wie bei dem Test festgestellt wurde, kann dieses Feature für diese Anwendung daher auch nicht sinnvoll eingesetzt werden.

5 Technische Dokumentation

Das Ziel bei der technischen Umsetzung der Applikation war es, die Funktionalität der iOS - Bestandsapplikation zu übernehmen, sowie die neu gewonnen Funktionen des Android Betriebssystems zu nutzen, um die Interaktionsoberfläche zu analysieren und gefundene Mängel zu verbessern. Im folgenden Abschnitt wird die technische Umsetzung der Applikation näher erläutert.

5.1 Programmierdokumentation

Der größte Teil der Dokumentation befindet sich innerhalb der entwickelten Klassen. Dies gelang dadurch, dass Codestücke ausführlichst mit Javadoc dokumentiert wurden, sowie sprechende Namen für Klassennamen als auch Variablen gewählt wurden. Durch das Formatieren von Kontrollstrukturen, welches das Ein- und Ausrücken inkludiert, wird der geschriebene Quellcode übersichtlich und lesbar.

5.2 Anforderungsdokumentation

Unter Android gibt es ein breitgefächertes Spektrum an verfügbaren Versionen, die auch mit einer unterschiedlichen Nutzeranzahl versehen sind. Ziel ist es eine Applikation zu entwickeln, die eine möglichst hohe Benutzeranzahl erreichen kann. Aufgrunddessen wurde das API Level 9 ausgewählt, da somit alle Versionen ab Android 2.3 unterstützt werden. Dieser Schritt wurde aufgrund folgender statistischer Erhebung gewählt. Mit dieser Konfiguration wird eine Benutzeranzahl von knapp über 99 Prozent erreicht.

Version	Codename	API	Distribution
2.2	Froyo	8	0.8%
2.3.3 - 2.3.7	Gingerbread	10	14.9%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	12.3%
4.1.x	Jelly Bean	16	29.0%
4.2.x		17	19.1%
4.3		18	10.3%
4.4	KitKat	19	13.6%

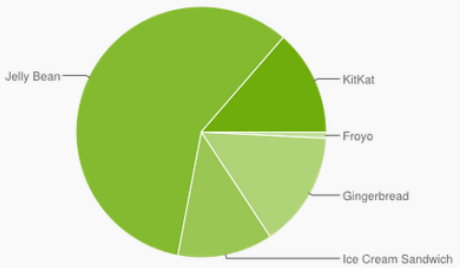

Abbildung 28. Stastische Nutzungsverteilung der Android-Versionen [11]

5.3 Installationsdokumentation

Um die Applikation installieren zu können, wird lediglich ein Endgerät benötigt, welches auf dem Betriebssystem Android basiert. Die Applikation wird ab Version 2.3 von jedem Android Gerät unterstützt.

Nach erfolgreichem Download und der nachfolgenden Installation aus dem Google Play Store kann die Applikation über den Home-Screen des Android Geräts gestartet werden.

5.4 Anwenderhandbuch

Nach der Installation startet man die Applikation mit einem Klick auf das Icon am Home-Screen des Android Geräts.

Zuerst befindet man sich im Hauptmenü der Applikation. Die Applikation bietet die Möglichkeit aus fünf verschiedenen UML-Diagrammen auszuwählen. Sie beinhaltet das Klassendiagramm, Sequenzdiagramm, Zustandsdiagramm, Aktivitätsdiagramm und das Anwendungsfalldiagramm. Im Hauptmenü (siehe Abbildung 29) gibt es am oberen Bildschirmrand ein Menü, um den aktuellen Status der gesamten Applikation zurückzusetzen, Informationen über die Entwickler

und das zugehörige Buch zu erfahren, die Möglichkeit ein Feedback an die Entwickler zu senden und den Fragenkatalog zu erneuern. Außerdem kann man die Farbeinstellungen des Feedback-Modus anpassen. Zur Auswahl stehen die Farben Rot und Grün, Blau und Gelb, Hell- und Dunkelgrau, sowie die Möglichkeit die gesamte Antwort mit der Farbe auszufüllen oder nur den linken Rand.

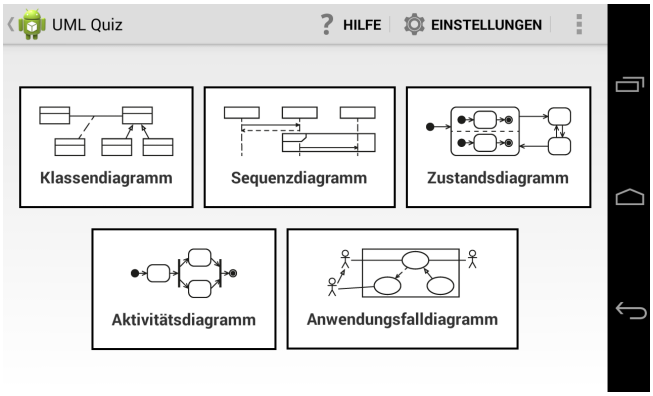

Abbildung 29. Ansicht des Hauptmenüs

Durch einen Klick auf einen UML-Diagrammtyp startet man ein Quiz. Wenn man die Applikation das erste Mal gestartet hat und noch keine Fragen heruntergeladen hat, so wird man aufgefordert ein Update durchzuführen. Für dieses Update wird eine Internetverbindung benötigt. Sobald das Update abgeschlossen, ist kann ein Quiz gestartet werden. Die Applikation verfolgt ein einheitliches Schema, welches wie folgt integriert ist.

Wenn man einen UML-Diagrammtyp anklickt, bekommt man die Möglichkeit ein vorhandenes Quiz fortzusetzen oder ein neues Quiz zu starten und das alte Quiz verfallen zu lassen.

Sobald ein neues Quiz gestartet wird, wird man gefragt, aus welcher Lernkartei die Fragen genommen werden sollen (siehe Abbildung 31). Das bedeutet, dass wenn bei einem vorherigen Quiz eine Frage richtig beantwortet wurde, wird diese um eine Kartei nach vorne versetzt. Falls man jedoch eine Frage in einer höheren Lernkartei falsch beantwortet, wird diese wieder in die erste Kartei zurückgelegt.

Durch einen Klick auf die jeweilige Kartei wählt man die darin liegenden Fragen

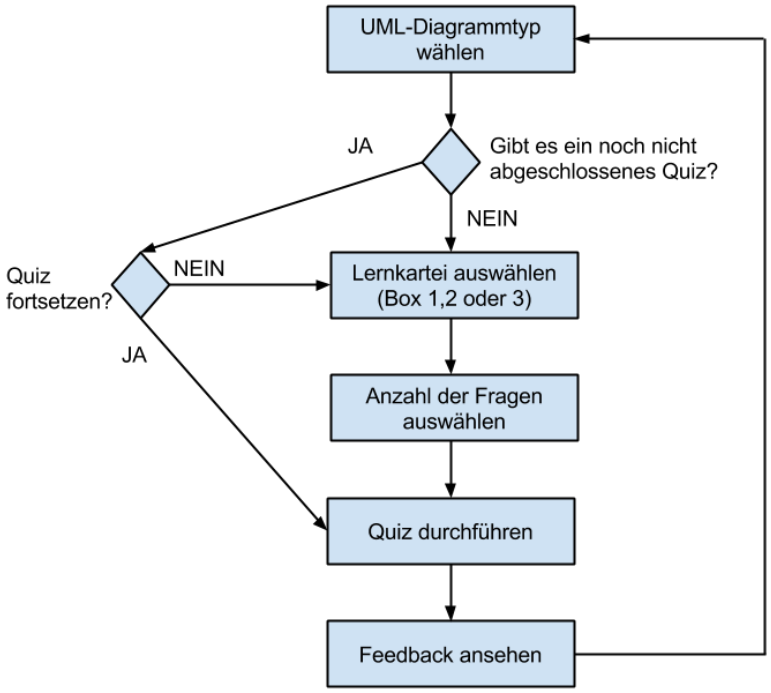

Abbildung 30. UML-Ablaufdiagramm der UML-Quiz Applikation

Abbildung 31. Ansicht der Lernkartei-Übersicht

für das nächste Quiz aus. Im nächsten Schritt muss man entscheiden, wieviele Fragen man für das Quiz laden möchte. Dies können zehn, zwanzig, dreißig oder alle enthaltenen Fragen sein (siehe Abbildung 32).

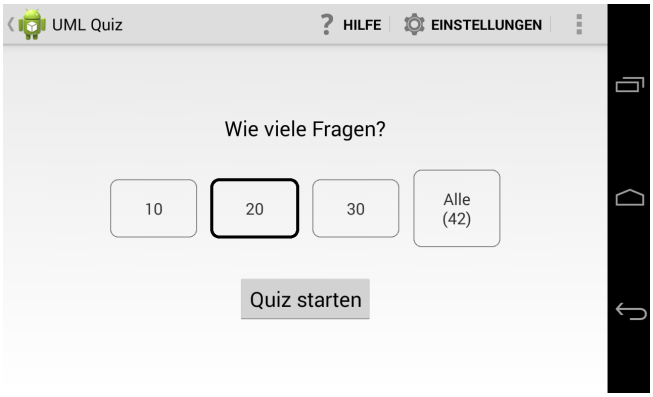

Abbildung 32. Ansicht des Auswahlmenüs für die Anzahl der Fragen

Nachdem die Anzahl der anzuzeigenden Fragen ausgewählt wurde, wird ein Quiz gestartet (siehe Abbildung 33). Durch das Wischen nach Links und Rechts wechselt man zwischen den Fragen. Mit einem Klick auf eine Antwort wird diese zwischenzeitlich als ausgewählte Antwort vermerkt. Dies ist deutlich erkennbar durch den dickeren Rahmen um die Antwort, sowie dem Häkchen auf der linken Seite. Es können immer eine oder mehrere Antwortmöglichkeiten zu einer Frage richtig sein. Eine Frage oder Antwort kann aus einem Text mit Bild, nur Text oder nur einem Bild bestehen.

Nachdem man die ausgewählte Anzahl an Fragen durchgewischt hat, kommt man zu einer Abfrage, ob man das Quiz beenden möchte und das Feedback einsehen will. Durch ein Wischen nach Links kehrt man zurück zu den Fragen und kann nochmals seine Antworten überdenken. Wenn man jedoch nach Rechts wischt, so bekommt man ein Feedback zu den ausgewählten Antworten und kann überprüfen, wo man Fehler gemacht hat und, wenn vorhanden auch ein textuelles Feedback zu einer Antwort lesen (siehe Abbildung 34). Ob ein Feedback-Text vorhanden ist, wird durch ein Fragezeichen am linken Rand der Antwortmöglichkeit gekennzeichnet. Richtig gewählte Antworten werden mit einem Häkchen versehen, während falsch gewählte Antworten mit einem Kreuz versehen sind. Wenn eine richtige Antwort vergessen wurde, als richtig zu markieren, so wird diese mit einem Rufzeichen versehen.

Sobald man die letzte Frage weggewischt hat, bekommt man eine Information, dass man das Quiz beenden kann. Das Quiz wird mit einer Wischbewegung

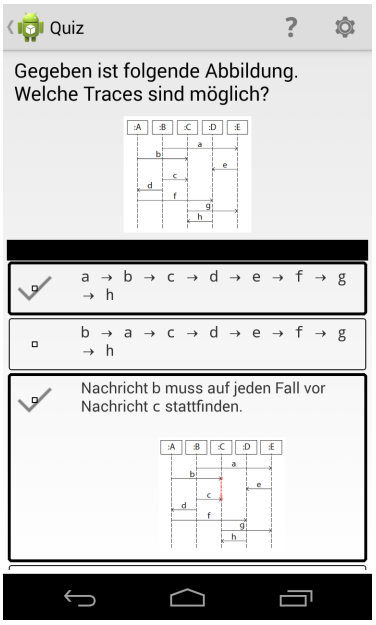

Abbildung 33. Ansicht der Fragen-Ansicht

nach rechts geschlossen und man findet sich wieder im Hauptmenü ein und kann ein neues Quiz starten.

5.5 Architekturbeschreibung

5.5.1 Datenbank - Schicht Die Applikation basiert auf einer SQLite Datenbank die im Android System automatisch integriert ist. Die Datenbank wird genutzt, um die fünf Typen von Diagrammen zu speichern, den Status von einem Quiz, die Auswahl der möglichen Fragen, die dazugehörigen Antwortmöglichkeiten und Beziehungen zwischen einem Quiz und den Fragen, sowie einem Quiz und den vom User ausgewählten Antworten.

Ein Quiz speichert die Informationen darüber, welche Fragen in diesem Quiz inkludiert sind und welche Antworten vom Benutzer der Applikation zu diesem Quiz ausgewählt wurden. Dies ermöglicht es, bei einem Neustart der Applikation zu wissen, welches Quiz von welchem UML-Diagrammtyp existent ist und welche Antwortmöglichkeiten, bei einem vorhergenden Versuch das Quiz abzuschließen, ausgewählt wurden.

Der Zugriff auf die Datenbank innerhalb der Applikation wird über sogenannte Content Resolver und Content Provider geregelt [3]. Der Content Resolver kümmert sich dabei, um die Anfragen der Clients und leitet diese an den zugehörigen Content Provider weiter. Der Content Resolver stellt die CRUD (Crea-

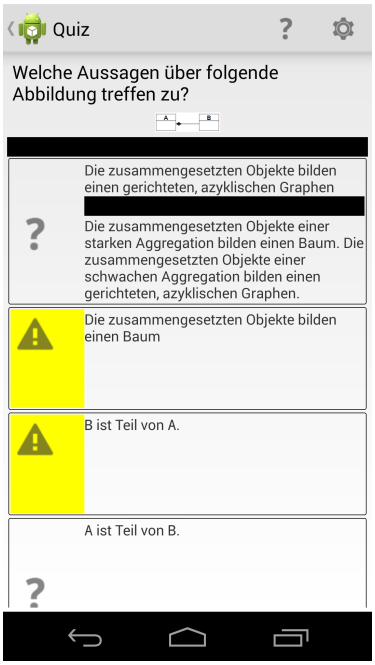

Abbildung 34. Ansicht der Feedback-Ansicht

te Read Update Delete) Methoden zum Zugriff und Verarbeiten von Daten zur Verfügung. Der Content Provider kümmert sich um die essentielle Datensicherheit, nämlich den Zugriff auf die Datenbank. Er gewährt nur dann Schreib- oder Lesezugriff, wenn dies ausdrücklich gewünscht wird. Er bildet sozusagen eine Abstraktionsebene zwischen der Datenbank und den zur Verfügung gestellten Daten. Dieses Pattern wurde deshalb gewählt, da es eine hohe Sicherheit im System gewährleistet und den Zugriff von anderen Applikationen nicht zulässt. Außerdem musste die Android-Support-Library eingebunden werden, da dieses Pattern nicht auf allen Android Versionen unterstützt wird, die unterstützt werden sollen. Aufgrund des Einsatzes der Support-Library ist dies jedoch ohne Probleme möglich.

Der Zugriff auf die Daten erfolgt unter Zuhilfenahme von Uri's. Eine Uri ist ein eindeutig identifizierender Aufruf. In diesem Fall werden eindeutig definierte Methodiken angesteuert, die in den verschiedenen Content Providern hinterlegt sind.

5.5.2 Business - Schicht Die Business-Schicht kümmert sich um die Zugriffe auf die benötigten Uri's und stellt die aufbereiteten Daten der GUI zur Verfügung. Durch diese Schicht schafft man ein höheres Abstraktionslevel und kann die Daten so aufbereiten, wie die darüber liegende GUI es benötigt.

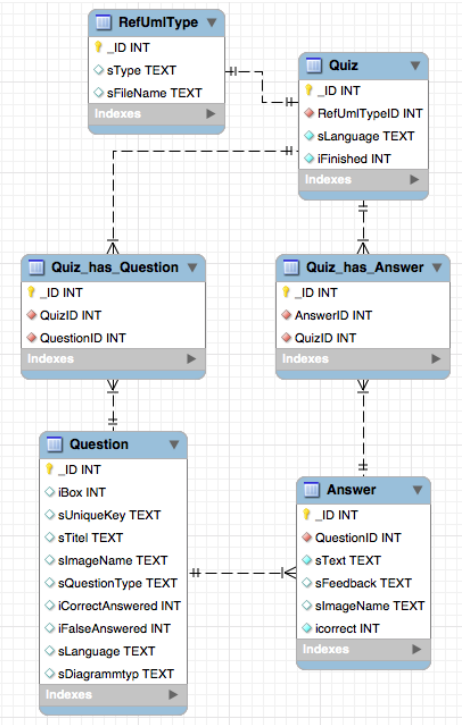

Abbildung 35. Datenbankstruktur des UML-Quiz

5.5.3 GUI - Schicht In der GUI-Schicht wird für jede einzelne Seite der Applikation eine Activity angelegt. Dadurch können offene Threads minimiert werden und die Funktion der Zurück-Taste ausgeschöpft werden. Durch die Trennung in einzelne Activities wird die Verwaltung der einzelnen Klassen einfacher und die Programmierrichtlinien können einfacher eingehalten werden, um einen sauberen und lesbaren Code zu erzeugen. Aufgrunddessen muss ein Aktivitätsdiagramm erstellt werden, um die Funktion der Zurück-Taste besser planen zu können.

5.5.4 XML - Schnittstelle Die Fragen für das Quiz stehen in Form eines XML-Files zur Verfügung. Dieses XML-File wurde aus dem TUWEL - System der TU Wien exportiert. Es beinhaltet drei wichtige Fragetypen bestehend aus dem Typ "multichoice" (Fragen mit mehreren Antwortmöglichkeiten), "matching" (Fragen mit genau einer möglichen Antwort) und "numerical" (Fragen die eine textuelle Antwort benötigen).

Grundlegend ist es nötig, dieses XML-File zu analysieren, um einen Parser zu schreiben, der das Dokument einliest und in die Datenbank speichert. Android bringt für diese Fälle den XmlPullParser zum Einsatz [10]. Basierend auf SAX

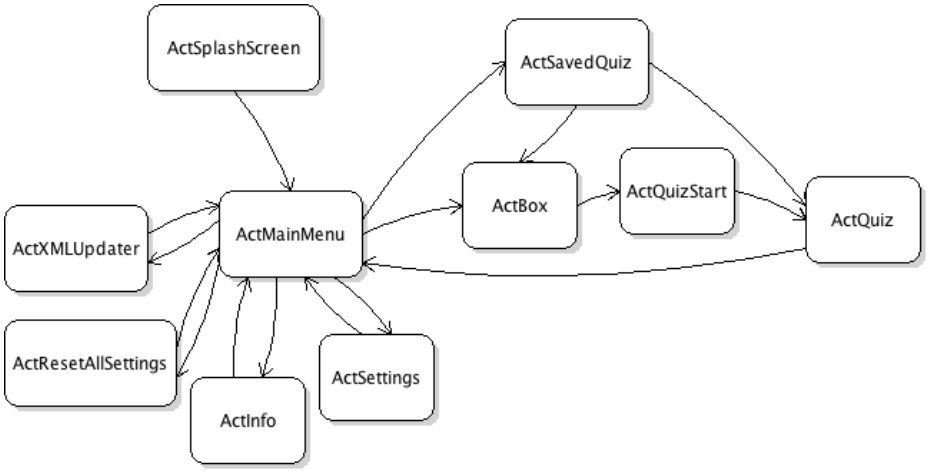

Abbildung 36. Activity - Flussdiagramm des UML-Quiz

liest der XmlPullParser einen Datenstrom vom Anfang bis zum Ende ein und analysiert während dem Einlesen die einzelnen möglichen Elemente. Das bedeutet, dass man nicht warten muss, bis eine Datei vollkommen eingelesen ist, sondern kann man Tag für Tag nach der Reihe auf die Elemente zugreifen über die DOM Architektur. Hierzu stellt der XmlPullParser die folgenden Events zur Verfügung, die aus dem START-TAG (Ein XML Start-Tag wurde gelesen), dem TEXT (Der Textinhalt zwischen einem öffnenden und schließenden Tag) und dem END-TAG (Ein XML End-Tag wurde gelesen) bestehen.

Als nächstes muss ein Primärschlüssel gefunden werden. Jede Frage besitzt einen eindeutigen Schlüssel, durch den diese eindeutig identifiziert wird. Dies ermöglicht es, bei Updates den Zusammenhang der Fragen festzustellen und eventuelle Updates darauf anzuwenden.

Ein weiterer kniffliger Punkt ist das Auslesen der Fragen und Antworten. Diese werden in unterschiedlichen Formaten zur Verfügung gestellt. Man unterscheidet hierbei reinen Text und Text im HTML-Format. Dadurch das Android HTML-Text unterstützt kann der Text einfachst in der GUI ausgegeben werden. Zuvor muss jedoch der img-Tag exportiert werden, da die Bilder nicht per HTML angezeigt werden sollen, sondern über Android selbst.

Das TUWEL System arbeitet mit Teilpunkten, jedoch die zu erstellende Quiz Applikation nicht. Aufgrunddessen wurden die Antworten beim Parsen so behandelt, dass Antworten mit einem positiven Anteil (fraction Attribut im XML-File) als richtig anzumerken sind und Antworten mit einem negativen Anteil als falsch zu deklarieren sind.

Der Fragentyp matching muss extra behandelt werden. Wenn die Fragen so geparst werden würden, wie das beim Multichoice der Fall ist, dann würde bei Fragen vom Typ matching nur die richtige Antwort angezeigt werden in der Applikation. Aufgrunddessen werden alle möglichen Antworten zu den einzelnen matching Fragengruppen ausgelesen und zu jeder einzelnen Gruppe von matching Fragen alle Antwortmöglichkeiten ausgegeben. Dies führt dazu, dass der matching Typ in einen multichoice Typ umgewandelt wird und das Quiz dadurch an Schwierigkeit gewinnt.

Ein Beispielcode für die genannte Thematik ist in Kapitel 7.2 ersichtlich.

5.5.5 Arbeitspakete der ersten Iteration In der ersten Iteration wird versucht, alle gewonnenen Erkenntnisse aus der Analysephase in Arbeitspakete zu fassen und umzusetzen.

Es wird ein Arbeitspaket für den XML-Import geben, der einen großen und wichtigen Teil der Applikation einnimmt, denn ohne Fragen gibt es auch kein Quiz. Zeitgleich muss die Datenbankstruktur aufgebaut werden, um die gewonnen Daten aus dem XML ablegen zu können.

Außerdem muss ein Aktivitätsdiagramm gezeichnet werden, um die benötigten Aktivitäten herauszufinden und wie diese miteinander vernetzt sind. Das Erstellen von Mockups für die einzelnen Aktivitäten und das Umsetzen der Layouts bildet ein weiteres Arbeitspaket.

Das wichtigste Paket mit dem größten Änderungscharakter ist das Arbeitspaket, welches sich mit der Funktionalität, sowie dem Design der Fragen beschäftigt. Hinzukommt das Entwerfen und Umsetzen des Feedback-Modus zu den Fragen.

Abschließend wird ein Arbeitspaket erstellt, welches ein einheitliches Design in die Applikation bringen soll. Die Aktivitäten der Applikation sollen mit ihrem Design aufeinander abgestimmt werden, um so das gewünschte einheitliche Design der NutzerInnen zu schaffen.

Nach der Umsetzung aller Arbeitspakete soll das Updaten von Fragen, das Durchführen von einem Quiz, das Beenden und erneute Starten von einem Quiz, sowie das Layout der Fragen- und Feedback-Aktivität vollständig funktionsfähig und ansehnlich gestaltet sein, um es den Nutzern für die abschließende Evaluierungsphase zur Verfügung stellen zu können.

6 Analyse und Erkenntnisse

Theoretische Grundlagen in Verbindung mit einem partizipativen Designprozess, die Bestrebung offizielle Designrichtlinien der Android Plattform umzusetzen,

konnten sehr gut dazu beitragen, eine entsprechend von den Testnutzern honorierte Lernapplikation zu entwickeln.

Aus Zeitgründen konnte vor Vollendung der Bachelorarbeit eine vollständige Hintergrundfunktionalität implementiert werden, die für die Evaluierung der Fragestellung ausreichend war, jedoch das komplette GUI-Interface nicht fehlerfrei implementiert werden.

Die Untersuchung, der von Android gebotetenen Accessibility-Features für sehschwache Personen, brachte einige bedeutende Designmängel zum Vorschein, die bei Accessibility bedacht werden müssen, bekräftigten aber auch, dass sich Software auf dem zu betreibenden System homogen integrieren muss, damit die vom Betriebssystem zur Verfügung stehenden Funktionen gut miteinander arbeiten können.

Viele theoretische Aspekte zeigten auch in der Praxis deren Gültigkeit. Dies gilt sowohl für das Design der Benutzeroberfläche als auch für die eigenständige durch den Anwender mögliche Erlernbarkeit der Bedienung.

Ungeachtet dessen wurde das UML-Quiz für Android, laut Auskunft der TestanwenderInnen, hervorragend auf Android portiert. Spezifische Features der Benutzeroberfläche und Interaktionsmöglichkeiten wurden entsprechend eingesetzt, so dass sich die Applikation auf der Android Plattform gut einbettet. Das Design und Bedienkonzept wurde weg von der vielfach kritisierten iOS-Applikation hin zu einer von den Nutzern gut akzeptierte Lernhilfe für das Studium entwickelt.

6.1 Konklusio des iterativen Entwicklungsprozesses

Aufgrund der beschränkten Zeit und dem Augenmerk auf Bedienbarkeit und Verständlichkeit für unbedarfte Nutzer ist es nicht gelungen eine Anwendung zu entwerfen, die besonders stabil läuft oder fehlerfreie Dialoge und Layouts auf einigen unterschiedlichen Geräten zu bieten. Dennoch kann die Anwendung auf den Entwicklungsgeräten fast störungslos dargestellt werden, das beinhaltet auch, dass Abstürze sehr selten auftreten. Für eine Evaluierung der in der Bachelorarbeit aufgestellten Fragen, reicht der Entwicklungsstand aber bei weitem aus.

6.1.1 Meinung der Entwickler Der iterative Entwicklungsprozess mit Einbindung der NutzerInnen hat dazu geführt, dass das UML-Quiz für Android eine Vielzahl an Verbesserungen der Benutzeroberfläche und Bedienbarkeit bietet. Auch konnte die Android Umgebung gezielt verwendet und ausgenutzt werden. Im Allgemeinen wird der Designprozess als gelungen empfunden. So konnten einige theoretische Ansätze erfolgreich umgesetzt werden, die von den Testnutzern begrüßt wurden. Durch das fortwährende Hinterfragen der Ideen konnten Designfehler frühzeitig erkannt und somit ausgebessert werden.

Wenige Kritikpunkte an der Anwendung richten sich, hauptsächlich weil diese vom zeitlichen Rahmen her nicht fertig gestellt werden konnten, an die Benutzeroberfläche. So wird das Hauptmenü oder die Karteikartenansicht nicht auf allen Geräten in vertikaler Positionierung einwandfrei dargestellt.

Dialoge und Beispielfragen beinhalten Rechtschreib- oder Kodierungsfehler. Rückmeldungen diesbezüglich wurden von der Evaluation ausgenommen. Die Benutzerauswahl für die Anzahl der Fragen entspricht nicht den Designrichtlinien. Ursprünglich wurden die Zahlen nur als Buttons angezeigt, bei denen der Nutzer keine Rückmeldung über die getroffene Auswahl erhalten hat. Daher wurden in nur kurzer Zeit eigene Buttons entwickelt, die auf dem Design der auszuwählenden Antworten beruhen und kombiniert mit der Funktionsweise einer Checkbox ausgestattet sind. Somit bekommt der Benutzer angezeigt, welche Auswahl getroffen wurde. Letztendlich befindet sich auf der Oberfläche noch ein Button im Stil einer Android Anwendung, der gar nicht dazu passt. Die Eigenentwicklung verstößt gegen das Prinzip einer homogenen Plattformoberfläche, trotzdem fügen sich diese in das individuelle Design der Anwendung ein, weil sowohl Hauptmenü oder Quiz-Ansicht keinerlei Standardsteuerelemente von Android nutzen.

Die Portierung auf Android ist als geglückt anzusehen, allerdings ist es möglich weitere Optimierungen bei den Activities vorzunehmen. So könnte die Frage, ob ein bereits gestartetes Quiz fortgeführt werden soll, in einer aufscheinenden Dialogbox beantwortet werden. Zumal die Auswahl der beiden Eigenschaften demselben Kontext zugehört. Die Anzahl der zu stellenden Fragen, könnte mit einem Schieberegler fixiert werden. Die Navigation über die Actionbar ist zum Großteil noch nicht funktional, begründet weil bei der zukünftigen Entwicklung der Anwendung die ein oder andere Activity entfernt wird. So ist die Navigation geplante Rückwärtsnavigation über das App-Icon theoretisch möglich. In der Praxis jedoch ist es nur mit einer strikten Hierarchie von Eltern-Activities lösbar. Eine dynamische Programierung ist zum derzeitigen Entwicklungsstand zu umständlich und erfolgt zu einem späteren Zeitpunkt.

Darüber hinaus ist es möglich, bei Android die Software für mehrere Sprachen zu entwickeln. Es sind nur zusätzliche Dateien mit entsprechenden Textwerten hinzuzufügen. Somit ist es kein Problem in naher Zukunft nicht nur die zur Verfügung stehenden englisch-sprachigen Quizfragen zu gebrauchen, sondern auch die passende Benutzeroberfläche für englisch-sprechende Nutzer mitzuliefern. Die Anwendungen müssen den Nutzer ermöglichen, dessen "kulturellen und persönlichen Eigenheiten" zu berücksichtigen [8] [1]. Auch der in den "101 - Guidlines" beschriebene Ansatz den deutsch-sprechenden Nutzern die Einstellungsmöglichkeit zu geben, ob die Software eine persönliche oder höfliche Anrede bei der Kommunikation mit dem Anwender gebrauchen soll [9].

6.1.2 Meinung der Testbenutzer Um die Lernförderlichkeit [8] des Android UML-Quiz zu evaluieren, wurden zwei Testpersonen im Alter zwischen 45 und 55 Jahren mit der Anwendung konfrontiert. Die Testpersonen verfügten über keinen Studienabschluss und waren wenig vertraut mit Android Geräten, hatten jedoch beruflich viel mit Programmen für komplexer Unternehmensstrukturen zu tun. Der Test erfolgte via E-Mail. Es wurde kurz erwähnt, dass es sich um eine Lern-Quiz Anwendung für Studenten handelt und eine Rückmeldung über die Bedienbarkeit und Benutzeroberfläche erwünscht ist. Für die Installation wurde die Binärdatei mitgesendet. Es war daher auch notwendig in wenigen Sätzen die Installation mit Android ("Installation aus Unbekannter Herkunft") zu beschreiben. Eine Testperson installierte das Programm am Smartphone, die andere am zehn Zoll Tablet.

Das Programm wurde in der Rückmeldung von beiden Personen als "sehr übersichtlich und leicht verständlich" gelobt, obwohl die Anwendung primär für Mobiltelefone entworfen wurde, war die Nutzung am Tablet als "sehr benutzerfreundlich" eingeschätzt worden. Negativ wurden die kleinen Symbole für die Erklärung der Steuerung bemängelt. Eine Testperson machte den Vorschlag die Icons bei den Farbeinstellungen ebenfalls zu erklären. Die andere empfahl den Text unterschiedlicher zu formatieren. So wurde konkret vorgeschlagen bei Frage-Sätzen, jeden einzelnen in eine Zeile zu schreiben und diese vielleicht in kursiv darzustellen. Dieser Vorschlag bestätigt den theoretischen Ansatz, dass mehrere Schriftarten die Differenzierung von Inhalten für das Gehirn erleichtern. Während der Softwareentwicklung wurde auch in diesem Zusammenhang die Schriftgröße des Fragetextes bereits erhöht (Funktioniert im UML-Quiz derzeit nicht auf allen Android Versionen), um die Inhalte besser voneinander abzugrenzen.

Zwei TestanwenderInnen im Alter zwischen 25 und 30 Jahren, die keinerlei Hochschulabschluss aufwiesen, fanden das Bedienkonzept der Anwendung verständlich. Diese beiden Nutzer erkundeten das UML-Quiz gemeinsam. Als Kritikpunkt wurde das Layout des Hauptmenüs (das zu dem Zeitpunkt noch nicht für unterschiedliche Geräte optimiert wurde) und zu klein dargestellte Bilder in den Antworten erwähnt. Hier wurde angeregt, die Bilder größer und scrollbar zu machen. In Anbetracht dessen, dass beide NutzerInnen kein Wissen über die fachliche Materie des Quiz besaßen, wollten die beiden von sich aus nicht mehr Kritik üben.

Hier zeigt sich, dass Usability Tests nicht nur durch Beobachtung alleine, sondern in manchen Fällen auch durch aktives Handeln des Testmoderators beeinflusst werden müssen, um ausreichend Rückmeldung erhalten zu können. Der Moderator muss in einer neutralen Weise in den Testprozess eingreifen und mit gezielten Fragen an den Benutzer Gedanken oder Verhaltensweise in Erfahrung bringen [23].

Eine weitere Testperson, die etwa 35 Jahre alt ist, Medieninformatik studiert und bereits 2D Spiele für Android entwickelt hat, konnte die Lehrveranstaltung für das UML-Quiz bereits abschließen. Der Test erfolgte ebenfalls via E-Mail. Ein Zitat beschreibt die positive Rückmeldung sehr gut: "Die App finde ich insgesamt sehr toll! Ich hätte sie gerne zum Üben der Diagramme gehabt." Positiv wurde die individuell mögliche Farbwahl, die Beschreibung der Steuerung mit Symbolen und intuitive Bedienung angesehen. Das Design des UML-Quiz wurde als entspannend und trotz des zahlreichen Inhalts als wenig überladen beurteilt. Als negativ betrachtet wurde die vertauschte Reihenfolge der Vorschaufelder von Rot-Grün bei der Einstellungsansicht. Die Grafiken wurden als zu klein empfunden. Als Wunsch wurde eine Übersicht der richtig und falsch beantworteten Fragen geäußert und auch die Möglichkeit nur fünf Fragen für eine Quizeinheit auszuwählen. Dieser Vorschlag kann gleich gesetzt werden, mit dem der Entwickler einen Schieberegler für die Anzahl der Fragen bereit zu stellen. Zusätzlich wurde die Hinterlegung einer nicht beantworteten Frage, die aber richtig gewesen wäre, mit der selben Farbe wie eine falsch beantworteten Frage als irritierend empfunden.

Diese Benutzermeinung wiederspiegelt die theoretische Tatsache, dass Farben und zierende Stilelemente sorgsam bis gar nicht eingesetzt werden müssen. Die Entscheidung die Frage von den Antworten nicht mit unterschiedlichen Farbtönen voneinander abzutrennen, hat sich diesem Résumé nach als richtig erwiesen. Der sparsame Einsatz von Farben und die Nutzung leichtgewichtiger Steuerelemente mit dezentem Rahmen geben der Anwendung ein homogenes Gesamtbild. Der Empfehlung des EVADIS II Standards, maximal 4 verschiedene Farben einzusetzen und doppelte Codierung (Rot-Grün, Blau-Gelb) anzubieten, kann Recht zugesprochen werden [25].

Die Idee, eine Übersicht der beantworteten Fragen zu erhalten, wurde im Gegensatz zur iOS Applikation in der Android Anwendung bewusst nicht umgesetzt. Folgende Gründe waren ausschlaggebend dafür: die Anzahl der Activities und somit die Anzahl der Benutzereingaben, als auch die Zeit, die für eine Quizrunde aufgewendet werden muss, zu verringern. All diese Kriterien sind ausschlaggebend für eine mobile Nutzung durch den Anwender.

Gerade der Wunsch auch nur fünf Fragen anzuzeigen, rührt wohl daher, dass man auch nur kurze Quiz-Einheiten absolvieren können sollte. Gerade bei so wenigen Fragen, machen Übersichtsseiten keinen Sinn, da der Nutzer weniger Zeit für das Durchgehen der geringen Anzahl an Fragen aufwendet, als würde dieser nachdenken müssen, welche Antwort von der Übersicht ausgewählt wird. Trotzdem wird an dieser Stelle angemerkt, dass keiner der TestnutzerInnen äußerte, den Fragestatus übersichtlich am Anfang der Frage anzuzeigen (bsp. Richtig, Falsch, teilweise Richtig). Diese Idee entstand den Softwareentwicklern während des Testprozesses.

Kritik zur Bedienung kam ebenfalls ausschließlich von einer Testperson. Sie merkte an, dass eine oder mehrere Antworten zu einer Frage richtig sein könnten. Der Fehler darf aus theoretischer Sicht nicht als grob angesehen werden, da die Benutzerin nach dem ersten Versuch ohnedies in der Lösung erfährt, dass mehrere Antworten richtig sein können [15]. In der Praxis wird den Studenten in der Vorlesung mehrmals mitgeteilt, wie eine Prüfung aussieht und zu beantworten ist. Analog dazu ist die Funktionsweise des UML-Quiz. Die Ansonsten von der Testperson intuitiv beschriebene Bedienung kann die gelungene Portierung auf Android und Integration in das Bedienkonzept des Betriebssystems bekräftigen. Die Erwartungskonformität und Erlernbarkeit der Anwendung ist dieser Meinung nach gegeben.

Wie im Falle der Icons muss auch die Verwendung von Sprache ähnlich hinterfragt werden. Bedingt dadurch, dass die Symbole für die Erklärung der Bedienung vielen Nutzern als zu klein empfunden wurden, war einem Testbenutzer der erwähnte Begriff "Swipe" gänzlich unbekannt und unverständlich, obwohl der 23-jährige Nutzer Erfahrung mit der Bedienung von Smartphones hatte. Allerdings war dieser weder Student, noch hatte dieser Erfahrung mit Softwareentwicklung. Dies ist der einzige Fall, bei dem unverständliche Sprache gefunden wurde. Das zeigt, dass sprachliche Elemente und Verständnis ebenso gelernt werden müssen wie die korrekte Assoziierung von Symbolen. Das Ergebnis ist von theoretischer Seite weniger erschreckend, da im Bezug nehmend zu den Icons die ISO/IEC CD 11581-1.2 Norm eine Kenntnis von zwei Drittel der NutzerInnen als ausreichend erachtet und in der Praxis nie ein 100 Prozent Verständnis erwartet werden darf. Dieser eine verwunderte Benutzer bestätigt paradoxerweise die Verständlichkeit der durchdachten Hilfestellung für Erstbenutzer, denn ohne diesem wäre die Problematik der sprachlichen Verständlichkeit außer Acht gelassen worden.

Außerdem wurden sieben weitere Testpersonen mit unterschiedlichem Bildungsstand befragt. Dies beginnt beim einfachen Studenten bis hin zu erfahrenen Personen in der Geschäftsführungsebene.

Einer dieser Nutzer hat in einem Review angegeben, dass er einen kurzen Erklärungstext auf der Startseite wünscht, denn der Nutzer stand vor einem Bildschirm mit fünf verschiedenen Bildern und einem Diagrammnamen, jedoch war es anfangs für ihn nicht sofort erkennbar, dass er durch einen Klick auf das Bild etwas machen kann.

Erstmalige NutzerInnen waren verwirrt, als sie ein Diagramm angeklickt hatten und drei Ordner angezeigt wurden, wobei in einem Ordner Fragen enthalten waren und in den anderen nicht. Hier sollte ein Text hinzugefügt werden, um was es sich dabei handelt.

Es wurde angemerkt, dass die Erklärung, wie ein Quiz funktioniert, sehr gut

dargestellt wird, jedoch wird es gewünscht beim Tutorial einen Pfeil nach unten darzustellen, um ersichtlich zu machen, dass man nach unten scrollen kann.

Letztendlich wurde auch zum Scrollverhalten der Applikation eine gegenteilige Meinung eingeholt. Es wurde angemerkt, dass es sinnvoller wäre, wenn die Frage immer oben stehen bleiben würde und nur die Antworten scrollbar wären. Zwar war sich der Nutzer der Thematik ersichtlich, dass dadurch viel Platz verschwendet wird, jedoch hat der Nutzer vorgeschlagen, dies mit einer Nutzergruppe auszutesten.

7 Anhang

Im Anhang findet man Interaktionsprotokolle mit den Nutzern und Code-Snippets, die zur näheren Erläuterung von einzelnen Kapiteln dienen.

7.1 Protokolle

Im Zuge der partizipativen Systemgestaltung werden Interviews mit den einzelnen Nutzern der Nutzergruppe geführt. Zur näheren Analyse wird Protokoll geführt, um im Nachhinein die Aussagen besser analysieren zu können. Ein hoher Detailierungsgrad kann erreicht werden.

7.1.1 Erste User Evaluierung Basierend auf Empfehlungen der Literatur wurde versucht, Fehler im derzeitigen Design der iOS-Applikation aufzufinden und eine verbesserte Umsetzung auf Android zu erreichen. Es wurden Skizzen für mögliche Layouts entworfen, dessen Diskussion mit potentiellen Benutzern der Software angestrebt wurde, um deren Rückmeldung in das weitere Design der Benutzeroberfläche einfließen lassen zu können.

Am Morgen des 7.4.2014 fand in den geschäftlichen Räumlichkeiten von Herrn Kletzander ein Treffen mit zwei Studentinnen statt. Studentin 1 studiert Medizin, und hat keinerlei Erfahrungen mit Softwareentwicklung und ist Anwenderin einer Lernplatform namens "Anki" auf einem Mobiltelefon.

Studentin 2 studiert Medieninformatik an der TU Wien und hat ihre bisherigen Erfahrungen in der Softwareentwicklung ausschließlich in Lehrveranstaltungen ihres Studiums erworben. Im Gegensatz zu den Verfassern der Bachelorarbeit hat Studentin 2 Freifächer und Softskillfächer belegt, die Sozi-Technische Punkte und Designpunkte in der Softwarentwicklung behandeln.

Den Studenten wurde zu Beginn mit wenigen Sätzen erklärt, dass eine mobile Lernplatform für die Lehrveranstaltung objektorientierte Modellierung zu entwickeln ist. Es werden Multiple-Choice Fragen mit einem Lernkartensystemm für die Prüfung der Lehrveranstaltung vorbereitet. Eine Applikation für iOS existiert bereits, diese soll für Android portiert werden.

Folglich wurden Screenshots der iOS-Applikation präsentiert. Dabei wurde hauptsächlich über das Layout diskutiert. Einstimmig wurde befunden, dass eine Aufteilung Frage/Antwort auf die Linke/Rechte Seite des Bildschirms nicht natürlich ist und vom Benutzer keinesfalls erwartet wird. Darüberhinaus können beide Hälften vertikal gescrollt werden, was die Usability zusätzlich komplizierter gestaltet. Dem Vorschlag die Frage und Antworten vertikal untereinander anzuordnen, wurde einstimmig zugestimmt. Studentin 1 verwies betreffend dem Layout auch auf eine von ihr genutzte Anwendnung Namens "Anki", bei der dies ebenso ist. Eine Diskussion über Anki erfolgt zu einem späteren Zeitpunkt, um den Fokus der Diskussion nicht zu verlieren. Auch wurde in den Raum gestellt, dass es wohl besser sei, ein Layout anzubieten, dass dem der realen Prüfung am ehesten entspricht. Dadurch wird der Benutzer besser auf diese vorbereitet und effizient die Prüfungsangst genommen. Antworten seien bei Prüfungen zumeist unterhalb der Prüfungsfrage.

Da auf mobilen Geräten weniger Platz für die Darstellung vorhanden ist, muss zwingend gescrollt werden. Dabei stellte sich die Frage, ob das Layout als ganzes vertikal oder lediglich die Antworten gescrollt werden können sollen. Hierbei hat sich die Meinung geschieden, jedoch führte Studentin 1 Anki vor. Bei Anki wird das ganze Layout gescrollt ohne, dass dies störend für sie als Benutzerin empfunden wird. Auch wurde als Argument angegeben, dass es mehr Zeit kostet zu tippen, dabei zusätzliche Inhalte dynamisch anzuzeigen, Inhalte zu vergößern oder verkleinern, als würde man als Benutzer scrollen. Es wurde von beiden Studentinnen als wichtig erachtet, nur mit einem Finger das ganze Layout in einer Richtung scrollen zu können. Bisher müssten nämlich Fragen und Antworten getrennt voneinander gescrollt werden. Dabei wurde erneut das Layout der iOS-Applikation kritisiert.

Es stellte sich zudem heraus, dass diese auch für Linkshänder nicht geeignet war. Da beim Betätigen diese den Finger über die Frage hinweg bewegen müssten. Die Sicht auf die Fragen ist dabei verdeckt. Der Weg, der zurückgelegt werden muss, verlängert sich und somit auch die Interaktionszeit. Um das Problem zu beseitigen, wurde von Studentin 2 vorgeschlagen, dazu eine Einstellungsoption anzubieten. Dies wurde aber, wie das horizontal gereihte Layout selbst, von allen anderen abgelehnt.

Die iOS-Applikation ist für eine horizontale Bedienbarkeit ausgelegt. Es wurde gefragt, ob dies ausreichend ist oder die Möglichkeit bestehen sollte die Anwendung auch vertikal zu bedienen. Es wurde erörtert, dass eine Anwendung für mobile Platformen selbstverständlich horizontale als auch vertikale Bedienbarkeit voraussetzen sollte. Eine horizontale Anwendung ist besser geeignet für mittelfristige Benutzung, wegen des menschlichen Blickfeldes. Dennoch ist auch die mobile Nutzung zu ermöglichen, die mit einer Hand und vertikal ausgerichtetem Mobiltelefon vonstatten geht. Dieser Argumentation wurde überwiegend

zugestimmt. Anwendungen für Mobiltelefone müssen für beide Layouts geschaffen sein, somit ist ein vom Benutzer zu erwartendes Verhalten gegeben.

Es wurde zuletzt das Anzeigen der Lösungen und Lernkarteiensystem zum Thema der Diskussionsrunde gemacht. Studentin 1 äußerte sofort Bedenken, wegen der Farbgestaltung und empfand nicht intuitiv, von der Darstellung, ablesen zu können, was richtig und falsch sei. Probleme für Personen mit Sehschwäche fanden auch Erwähnung und das simulierte Bild für Personen mit Rot-Blindheit (Protanopie) wurde hergezeigt. Studentin 1 meinte, es sei hilfreich genug, wenn nur richtige Antworten grün hinterlegt werden. Beide Studentinnen aber befanden es für notwendig, die Merkmale der Antworten mit Symbolen kenntlich zu machen. Auch sollten nur abstrakte Symbole (Hakerl, Kreuz) zum Einsatz kommen, weil Emoticons missverstanden werden oder zu schlechter Laune führen könnten. Letzteres für den Fall, dass der Benutzer für falsche Fragen traurige Emoticons angezeigt bekommt.

Individuelle Beschreibungen zu den Lösungen werden bei der iOS-Applikation dauerhaft angezeigt. Die Mehrheit befand, dass die Oberfläche überladen wirkt und der Benutzer unnötig scrollen muss. Es wurde daher vorgeschlagen, Beschreibungen zu einzelnen Antworten explizit mittels Benutzereingabe anzuzeigen und diese vorerst ausgeblendet zu belassen. Als Problem erweist sich dabei die Tatsache, dass nicht zu jeder Antwortmöglichkeit Beschreibungen existieren müssen. Als Kompromiss betreffend Design und Usability ist es für diesen Fall unerlässlich, die Existenz mittels Symbol darzustellen.

Das Lernkartensystem wurde von allen Diskussionsteilnehmern positiv angesehen. Dieses System ist in Österreich bei jungen Erwachsenen bekannt, da das Lernen für die theoretische Führerscheinprüfung am Computer mittels einer auf diesem Prinzip basierenden Software unterstützt wird. Studentin 1 merkte an, dass dieses Feature bei ihrer Anki-Anwendung fehlt, aber ihr sehr gelegen komme und wünschenswert wäre. Dabei ist ihr wichtig zu wissen, wieviele Fragen noch nicht richtig beantwortet wurden. Diese Aussage bekräftigt die Nachfrage für Auswertungen, nach statistischen Gesichtspunkten, der Anwendung für eine Rückmeldung des Lernerfolgs an den Benutzer.

Studentin 1 würde sich auch eine Funktion erhoffen, bei der irrtümlich falsch beantwortete Fragen im Nachhinein als richtig markiert werden können. Dieser Vorschlag wurde kritisch hinterfragt. Einerseits solle der Benutzer selbständig entscheiden können, andererseits könnte dies dem Benutzer dazu verleiten die Fragen voreilig als ausreichend gelernt zu befinden.

Zusammenfassung der gelösten Fragestellungen

Ist es angenehmer Frage und Antworten untereinander statt nebeneinander darzustellen?

JA, weil der Fragetext besser über den Bildschirm verteilt wird. Antwortmöglichkeiten können mit beiden Händen erreicht werden. Das gesamte Layout kann mit einer Hand gescrollt werden kann.

Soll die Bedienung horizontal als auch vertikal möglich sein?

JA! Dies wird vom Benutzer bei Anwendungen auf Mobilgeräten vorausgesetzt um in unterschiedlichen Situationen eine einfache Bedienung und Halteposition für das Gerät sicherzustellen.

Können die Farben Rot und Grün zur Hinterlegung für falsche und richtige Antworten genutzt werden?

NEIN! Eine Unterscheidung ist für Personen mit Protanopie nicht möglich. Es ist aussreichend nur eine Farbe für einen besseren Kontrast zu verwenden. Stattdessen müssen Symbole eindeutig erkenntlich eingesetzt werden.

Bringt der Einsatz von Emoticons einen Mehrwert für den Benutzer?

NEIN! Diese werden möglicherweise missverstanden oder demotivieren den Benutzer.

Ist es wünschenswert ein Lernkartensystem einzusetzen?

JA! Der Lernerfolg ist damit nachvollziehbar. Der Benutzer kann gezielt falsch beantwortete Fragen intensiver üben. Das Lernkartensystem wird von befragten Personen positiv angesehen.

Sind Lösungen und Erklärungen zu den Fragen am Ende vollständig anzuzeigen oder muss der Benutzer explizit interagieren?

NEIN! Wegen des geringen Platzangebotes auf Mobilgeräten würde die Benutzeroberfläche überladen wirken. Zielführender ist es, dass der Benutzer wenige unklare Antworten auswählt, dessen Auflösung gewünscht ist.

7.2 XML Beispielcode

```
<question type="matching">
    <name>
      <text>CD_Aggregationen</text>
    </name>
    <questiontext format="html">
      <text>Ordnen Sie die Begriffe zu!</text>
    </questiontext>
    <image></image>
    <generalfeedback>
      <text></text>
    </generalfeedback>
    <defaultgrade>1</defaultgrade>
    <penalty>0</penalty>
    <hidden>0</hidden>
    <shuffleanswers>0</shuffleanswers>
    <subquestion>
      <text>Eine Aggregation ist eine spezielle Form der ...</text>
      <answer>
        <text>Assoziation</text>
      </answer>
    </subquestion>
    <subquestion>
      <text>Wie wird die starke Aggregation noch genannt?</text>
      <answer>
        <text>Komposition</text>
      </answer>
    </subquestion>
  </question>
```

Literatur

1. 90/270/EWG. https://rsw.beck.de/rsw/upload/EUArbR/68_EWG_RL_90_270.pdf. Zuletzt besucht: 23.06.2014.
2. Alexander, K. *Kompendium der visuellen Information und Kommunikation*. Springer Berlin Heidelberg, Berlin, Heidelberg, Berlin, Heidelberg, 2007.
3. Android Design Pattersn. http://www.androiddesignpatterns.com/2012/06/content-resolvers-and-content-providers.html. Zuletzt besucht: 03.06.2014.
4. Anki. http://ankisrs.net/. Zuletzt besucht: 03.06.2014.
5. BRAINYOO. http://www.brainyoo.de/karteikarten-app/. Zuletzt besucht: 03.06.2014.
6. CoboCards. http://www.cobocards.com/de/. Zuletzt besucht: 03.06.2014.
7. Connect. http://www.connect.de/testbericht/ios7-android-4-windows-phone-8-vergleich-1907992.html. Zuletzt besucht: 16.06.2014.
8. EN ISO 9241-10. http://www.interactive-quality.de/site/DE/int/pdf/ISO_9241-10.pdf. Zuletzt besucht: 23.06.2014.
9. Furm-Hazivar, A. 101 guidelines; usability evaluation eines webbasierten e-learning-systems, 2010. Parallelt. [Übers. des Autors] 101 Guidelines: Usability Evaluation of a webbased E-Learning-System; Wien, Techn. Univ., Dipl.-Arb., 2010.
10. Google. http://developer.android.com/reference/org/xmlpull/v1/XmlPullParser.html. Zuletzt besucht: 03.06.2014.
11. Google. https://developer.android.com/about/dashboards/index.html?utm_source=ausdroid.net. Zuletzt besucht: 03.06.2014.
12. Google. http://www.bvdw.org/mybvdw/media/download/our-mobile-planet-germany-de.pdf?file=2266. Zuletzt besucht: 18.05.2014.
13. Google Android. http://developer.android.com/tools/testing/testing_accessibility.html. Zuletzt besucht: 03.06.2014.
14. Hammer, N. *Mediendesign für Studium und Beruf; Grundlagenwissen und Entwurfssystematik in Layout, Typografie und Farbgestaltung*. X.media.press. Springer Berlin Heidelberg, Berlin, Heidelberg, Berlin, Heidelberg, 2008.
15. Hobmair, H. H. *Pädagogik*, 3. aufl., korrigierter nachdr. ed. Bildungsverl. EINS, Stam, Troisdorf, © 2002; 2002.
16. Huang, A. H. Exploring the potential effects of emoticons. *Information & Management, 2008, Vol.45(7), pp.466-473* (2008). Instant messaging (IM) has shown signs of becoming one of the main stream communication applications for users, like e-mail. Many people maintain constant contacts with multiple friends and relations via IM simultaneously whenever they are online, whether working on other applications or not. In addition to allowing instant exchange of text information, a unique feature of IM is its use of graphical icons that express emotions, known as emotional icons or emoticons. We explored their potential effects. Our model, based on prior theory and research, was tested using data collected from student users; it was analyzed to reveal potential effects of emoticons on various factors related to the use of IM. Our study used structural equation modeling (SEM) analysis; the results showed that the user of emoticons felt a positive effect on enjoyment, personal interaction, perceived information richness, and perceived usefulness. Our results suggested, however, that emoticons were not just enjoyable to use, but also a valuable addition to communication methods.

17. Luke Wroblewski. Touch gesture reference guide. Zuletzt besucht: 17.05.2014.
18. Marcus Hegner. Methoden zur evaluation von software. Zuletzt besucht: 18.05.2014.
19. Marion Brandl. http://www.provinz.bz.it/sozialwesen/download/Emotion.pdf. Zuletzt besucht: 18.05.2014.
20. Noyes, Jan ; Baber, C. *User-centred design of systems*. Applied computing. Springer, London [u.a.], 1999.
21. Repetico. http://www.repetico.de/. Zuletzt besucht: 03.06.2014.
22. Rossmann, P. *Einführung in die Entwicklungspsychologie des Kindes- und Jugendalters*, 6. aufl., korrigierter nachdr. ed. Verlag Hans Huber, Bern, © 2010.
23. Rubin, J. D. C. Handbook of usability testing : how to plan, design, and conduct effective tests, 2008.
24. Ruminski, J.; Wtorek, J. Color transformation methods for dichromats. Color blindness is a serious perception problem. Suffering individuals cannot understand messages, which are carrying by many images, especially those, distributed by WWW. In this paper we are proposing three image processing methods to enhance image recognition and understanding by persons with dichromacy. Color difference image is introduced to represent color perception dissimilarity. Two color transformation methods are presented. Results of tests proved the usefulness of the methods for fast color transformation of images in the WWW environment.
25. Sarodnick, Florian ; Brau, H. *Methoden der Usability Evaluation; wissenschaftliche Grundlagen und praktische Anwendung*, 1. aufl. ed. Praxis der Arbeits- und Organisationspsychologie. Huber, Bern, 2006.
26. Stapelkamp, T. *Screen- und Interfacedesign; Gestaltung und Usability für Hard- und Software*. X.media.press. Springer Berlin Heidelberg, Berlin, Heidelberg, Berlin, Heidelberg, 2007.
27. Technologizer. http://www.technologizer.com/2011/10/31/the-one-really-nice-thing-about-androids-back-button. Zuletzt besucht: 18.05.2014.
28. TU-Berlin. http://www.uselab.tu-berlin.de/wiki/index.php/Partizipatives_Design. Zuletzt besucht: 10.06.2014.
29. von Franqué, A. Optimizin quiz-centered e-learning content for usability on mobile devices. Master's thesis, Vienna University of Technology, 2013.