

Masterarbeit

Group Decision Support with Mobile Devices

Philipp Leitner BSc

Institut für Softwaretechnologie

Technische Universität Graz

Vorstand: Univ.-Prof. Dipl.-Ing. Dr. techn. Wolfgang Slany



Betreuer: Univ.-Prof. Dipl.-Ing. Dr. techn. Alexander Felfernig

Graz, 2014

Abstract

Within the framework of this thesis, a domain-independent recommender system was adapted for mobile devices. The combination of the upcoming research area of group decision support systems and the expanding market of mobile devices and associated applications is the innovative aspect of this paper.

The focus during the development of the application named *WeDecideMobile* is on the usability and the intuitive operation of the user interface. The behavior driven development assures a high quality production code. The cross-platform implementation provides the availability for all popular mobile platforms.

Kurzfassung

Im Rahmen dieser Arbeit wurde ein domänenunabhängiges Gruppenempfehlungssystem für mobile Geräte adaptiert. Die Verbindung des aufstrebenden Forschungsgebietes von Systemen zur Gruppenentscheidungsunterstützung sowie dem wachsenden Markt der mobilen Geräte und dazugehöriger Applikationen stellt das Novum dieser Arbeit dar.

Der Fokus bei der Entwicklung der Applikation mit dem Namen *WeDecideMobile* liegt auf der Benutzerfreundlichkeit und einem intuitiven Benutzerinterface. Der verhaltensgetriebene Entwicklungsansatz stellt eine hohe Qualität des Produktionscodes sicher. Durch die plattformübergreifende Umsetzung konnte die Verfügbarkeit der Applikation für alle gängigen mobilen Plattformen sichergestellt werden.

Statutory Declaration

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

Eidesstattliche Erklärung

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

Graz, am

.....

(Unterschrift)

Danksagung

Zu Beginn möchte ich mich bei meinem Betreuer Herrn Univ.-Prof Dipl.-Ing Dr.techn. Alexander Felfernig bedanken, der mich in der letzten Phase meines Studiums unterstützt und geleitet hat.

Mein besonderer Dank gilt meiner Frau Martina, die mir während meines Weges beigestanden und auch in schwierigen Zeiten starken Rückhalt gegeben hat.

An meinen Sohn, Alexander ein großes Dankeschön! Danke, dass du mir immer wieder mit einem Lächeln die Arbeit versüßt hast.

Ein herzliches Dankeschön geht an meine Eltern für ihre bedingungslose und liebevolle Unterstützung in allen Lebenslagen.

Mein ganz besonderer Dank geht an meinen Bruder, der mir bei dieser Masterarbeit sehr hilfreich zur Seite gestanden hat und aufgrund seiner langjährigen Erfahrungen eine große Hilfe war.

Zum Schluss möchte ich noch ein herzliches Dankeschön an alle Studenten aussprechen, die ich während meiner Studiendauer kennen und schätzen gelernt habe.

Graz, im März 2014

Philipp Leitner

Inhaltsverzeichnis

1	Einleitung	2
2	Stand der Forschung	6
2.1	Relevante Gruppenempfehlungssysteme	7
2.1.1	Musik	7
2.1.2	Film und Fernsehen	8
2.1.3	Essen, Reisen und Tourismus	9
2.1.4	Andere und domänenunabhängige Gruppenempfehlungssysteme . .	10
2.2	Klassifikation von Gruppenempfehlungssystemen	13
2.3	Aggregationsmethoden	16
2.4	Plattformübergreifende Entwicklung	17
3	Systemarchitektur	19
3.1	Aufbau des Systems	20
3.1.1	Model View Controller	20
3.1.2	Schnittstellenübersicht	22
3.2	Verwendete Technologien	23
3.2.1	Apache Tomcat	23
3.2.2	Java Servlets	23
3.2.3	Hibernate	24
3.2.4	MySQL	25
3.2.5	JavaScript	25
3.2.6	JSON und JSONP	26
3.2.7	Sencha Touch	27
3.2.8	Apache Cordova	29

3.2.9	Jasmine	29
3.3	Entwicklung	30
3.3.1	Testgetriebene Entwicklung	30
3.3.2	Verhaltensgetriebene Entwicklung	32
4	Grafische Benutzeroberfläche	33
4.1	Hauptseite	34
4.2	Detailansicht	36
4.3	Benutzerbereich	39
4.4	Ablauf der Gruppenentscheidungsaufgaben	42
4.5	Antworten finden	44
4.5.1	Modellierung	44
4.5.2	Teilnahme	50
4.5.3	Finalisierung	52
4.6	Wähle eine Alternative	54
4.6.1	Modellierung	54
4.6.2	Teilnahme	56
4.6.3	Finalisierung	58
4.7	Suche & wähle eine Alternative	60
4.7.1	Modellierung	60
4.7.2	Teilnahme	63
4.7.3	Finalisierung	65
4.8	Einen Termin finden	66
4.8.1	Modellierung	66
4.8.2	Teilnahme	69
4.8.3	Finalisierung	71
4.9	Anforderungen priorisieren	73
4.9.1	Modellierung	73
4.9.2	Teilnahme	76
4.9.3	Finalisierung	77
4.10	Evaluierung	79
5	Zusammenfassung	80

6 Offene Forschungsfragen	82
Abkürzungsverzeichnis	87
Literaturverzeichnis	89

1 Einleitung

Viele der heutzutage auftretenden Entscheidungen im privaten oder ökonomischen Bereich werden nicht mehr von Einzelpersonen sondern von Familien, Gruppen, Teams oder Kommissionen gefällt [Jameson und Smyth, 2007; Masthoff, 2011]. Dies bringt einige Vorteile mit sich. Eine einzelne Person verfügt in der Regel nicht über das gesamte Wissen um eine Entscheidung optimal zu fällen. Die daraus resultierenden, möglicherweise fehlerhaften Einschätzungen können durch das Kollektiv ausgeglichen werden. Des Weiteren sind mehrere Personen durch ihre unterschiedlichen Betrachtungsweisen in der Lage, mehr Ideen und Lösungsvorschläge zu generieren. Weiters können durch den Dialog miteinander Argumente verhärtet aber auch revidiert werden [Aronson, Wilson und Akert, 2005]. Hinzu kommt der psychologische Effekt, denn Mitglieder, die an einer Entscheidungsfindung direkt beteiligt sind, fühlen sich dem Ergebnis mehr verpflichtet [Herr, Rösch, Beckmann und Gross, 2012].

Gruppenentscheidungsfindungen stellen aber auch einige Herausforderungen dar: Unter anderem der höhere Zeitaufwand um eine Entscheidung zu fällen, Gruppenpolarisierung bei der die Ansichten von Personen nach einem Dialog und Informationsaustausch extremer sind als zuvor oder Rivalitäten und Machtkämpfe in der Gruppe, die bei offenem Meinungsaustausch die Entscheidungsfindung erheblich erschweren. Um diese negativen Eigenschaften zu minimieren und die positiven zu verstärken bietet sich der Einsatz von Gruppenempfehlungssystemen an.

In den letzten Jahren waren Gruppenempfehlungssysteme im Fokus der Forschung innerhalb der Empfehlungssysteme, wodurch einige bemerkenswerte Systeme entwickelt wurden. Zur Auswahl eines passenden Musikkanals wurde 1998 *MusicFX* [McCarthy und Anagnost, 1998] entwickelt, für Gruppenempfehlungen im Filmbereich gibt es *Polylens* [O'Connor, Cosley, Konstan und Riedl, 2001] und für die Planung eines gemeinsamen Urlaubs kann

Collaborative Advisory Travel System [McCarthy, Salamó, Coyle, McGinty, Smyth und Nixon, 2006] eingesetzt werden.

Alle diese Systeme beschränken ihre Empfehlungen auf eine Domäne. Es gibt jedoch auch bereits Forschungsarbeiten im Bereich der domänenunabhängigen Gruppenempfehlungen [Garcia, Pajares, Sebastia und Onaindia, 2012]. Besonders hervor getan hat sich dabei das System mit dem Namen *WeDecide*, das fünf verschiedene Arten von Gruppenentscheidungsaufgaben anbietet, durch die die Benutzer ihre persönlichen Aufgaben modellieren können.

Die ursprüngliche Version der *WeDecide*-Umgebung besteht nur aus einem Webauftritt. Das Ziel dieser Arbeit, die Entwicklung einer mobilen Applikation zur Kommunikation mit dem *WeDecide*-Umgebung bietet sich an, da der Smartphone- und Tabletmarkt in den letzten Jahren mit rasanter Geschwindigkeit gewachsen ist.

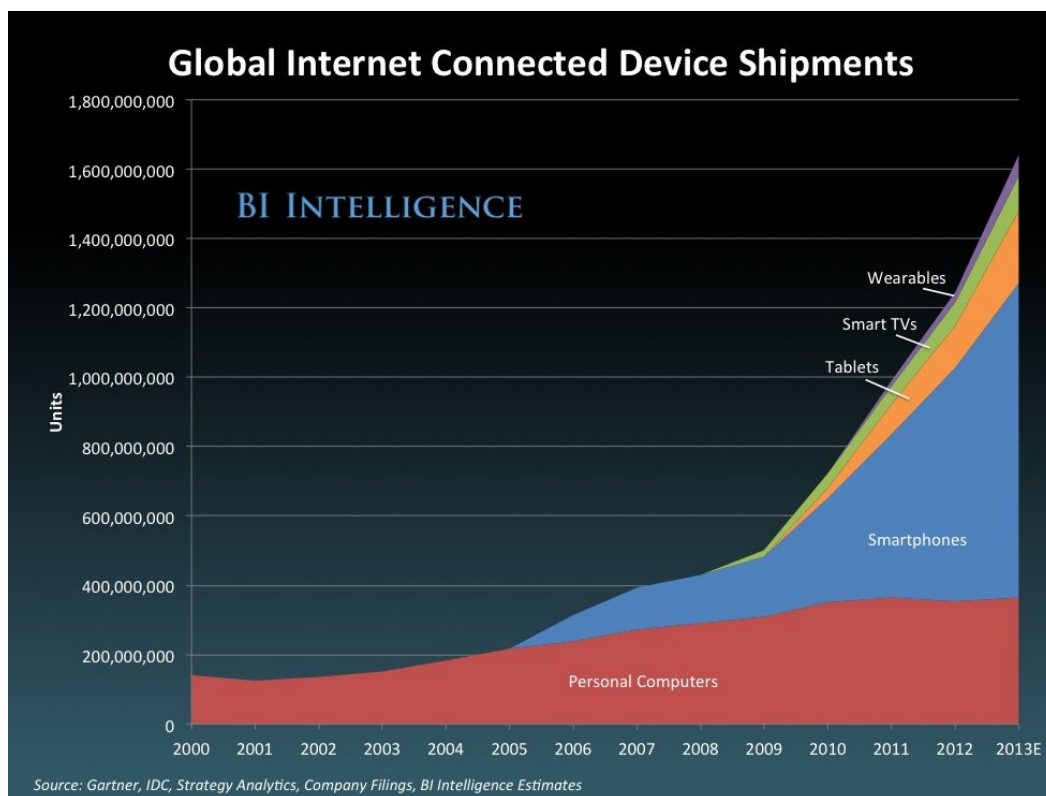


Abbildung 1.1: Global Internet Connected Device Shipments
[Worldwide Smart Connected, 2013]

Laut International Data Corporation (IDC), einem international tatigen Marktforschungs- und Beratungsunternehmen auf dem Gebiet der Informationstechnologie und der Telekommunikation, wurde die Milliarden­grenze der weltweiten Lieferungen von internetfahigen Gerate 2012 berschritten [Worldwide Smart Connected, 2013], wie aus der Abbildung 1.1 abzulesen ist.

Im Jahr 2013 konnte ein weiteres Wachstum von 7,2% gegenber dem Vorjahr verzeichnet werden und der Fnf-Jahresausblick auf 2017 prognostiziert eine Steigerung auf 2,3 Milliarden befrderte Smartphones im Jahr [Worldwide Mobile Phone, 2013]. Auch wenn die Betriebssysteme die bei diesen Geraten zum Einsatz kommen von den Plattformen Android und iOS dominiert werden, zeigt die Prognose fr 2017 einen Anstieg von Windows Phone und eine allgemein starke Diversifizierung (siehe Abbildung 1.2).

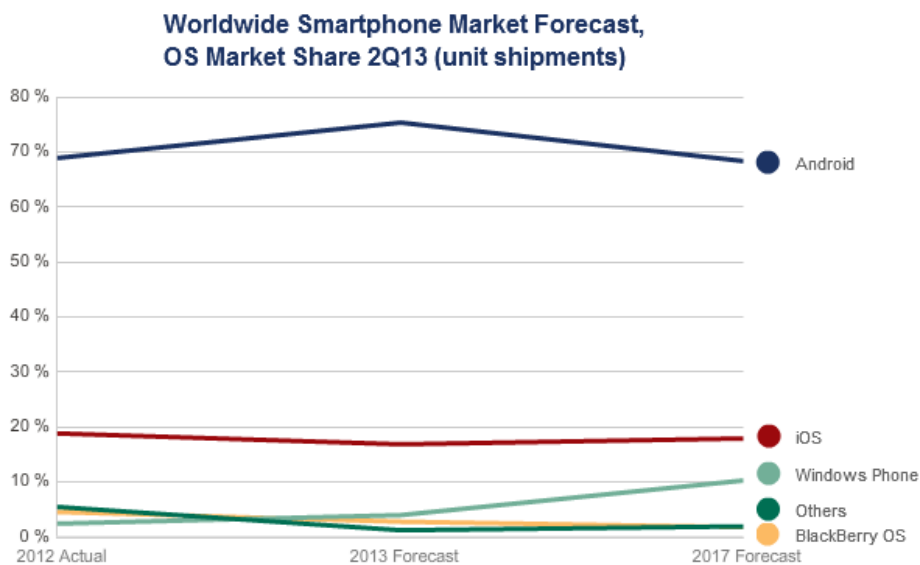


Abbildung 1.2: Weltweite Smartphone-Marktprognose 2013-2017
[Worldwide Mobile Phone, 2013]

Die Entwickler von mobilen Applikationen stehen dadurch vor der Wahl, sich entweder auf bestimmte Betriebssysteme einzuschranken oder den Mehraufwand fr jede der Plattformen in Kauf zu nehmen. Als Ausweg bietet sich die plattformbergreifende (engl. cross-plattform) Entwicklung einer Hybrid-Applikation an. Dabei wird im Gegensatz zur nativen Entwicklung

ein gemeinsamer Code geschrieben, aus dem dann die unterschiedlichen nativen Applikationen erstellt werden. Nach einer Evaluierung der vorhandenen Technologien wurde für diese Arbeit das Sencha Touch Framework verwendet.

Diese Arbeit ist neben dieser Einleitung in fünf Teilbereiche unterteilt. Den Anfang macht Kapitel 2 mit dem *Stand der Forschung*, in dem ähnliche Arbeiten und Projekte im Bereich der Gruppenempfehlungssysteme behandelt werden. Kapitel 3 *Systemarchitektur* beinhaltet die Architektur des Projekts, beschreibt die verwendeten Technologien, erörtert die implementierte Schnittstelle und erklärt den Ansatz des verwendeten testgetriebenen Entwickelns (engl. Test Driven Development). Mit der Beschreibung des grafischen Benutzerinterfaces befasst sich das Kapitel 4 *Grafische Benutzeroberfläche*. Den Abschluss bilden das Kapitel 5, das die Arbeit nochmals rückblickend zusammenfasst und das Kapitel 6, das Ansätze für zukünftige Arbeiten bietet.

2 Stand der Forschung

Die immer größer werdende Vielfalt von Produkten und Angeboten führte zur Entwicklung von Empfehlungssystemen (engl. Recommender Systems) [Cho, Kim und Kim, 2002; Min und Han, 2005; Resnick, Iacovou, Suchack, Bergstrom und Riedl, 1994], die zur Unterstützung der Entscheidungsfindung der Benutzer dienen. Diese Empfehlungsdienste bieten den Benutzern Vorschläge zu möglicherweise interessanten Produkten in den unterschiedlichsten Bereichen an. Dazu gehört die Empfehlung von passenden Restaurants [Burke, Hammond und Young, 1996], von richtigen Filmen [Miller, Albert, Lam, Konstan und Riedl, 2003] und vielen e-Commerce Produkten [Schafer, Frankowski, Herlocker und Sen, 2007].

Der Fokus lag lange Zeit auf den Empfehlungen für individuelle Nutzer. Es kommt jedoch immer öfter vor, dass nicht nur ein Individuum sondern auch eine Gruppe solche Systeme in Anspruch nehmen möchte. Dafür wurden Gruppenempfehlungssysteme (engl. Group Recommender Systems) entwickelt, die neben Empfehlungen für Einzelpersonen auch noch einige weitere Aufgaben zu lösen haben. Es gibt heutzutage bereits die unterschiedlichsten Arten von Gruppenempfehlungssystemen für die verschiedensten Domänen. Diese erstrecken sich vom multimedialen Bereich über Empfehlungen von Orten von Interesse (OVI) bis zu Empfehlungen im professionellen Bereich wie der Softwareentwicklung, bei der es vor allem um die Priorisierung von Anforderungen [Felfernig, Zehentner, Ninaus, Grabner, Maalej, Pagano, Weninger und Reinfrank, 2011] geht.

Das Kapitel 2.1 behandelt relevante Gruppenempfehlungssysteme, die in ihrer Domäne gegenübergestellt werden. Mit der Klassifikation der Gruppenempfehlungssysteme geht es in Kapitel 2.2 weiter, dabei werden Verweise zu den zuvor aufgezählten Systemen gemacht. Kapitel 2.3 behandelt die Strategien zur Aggregation der Informationen, die bei den Systemen zum Einsatz kommen. Das Kapitel 2.4 beschäftigt sich schließlich mit der Gegenüberstellung von plattformübergreifenden Entwicklungsmöglichkeiten für mobile Applikationen.

2.1 Relevante Gruppenempfehlungssysteme

2.1.1 Musik

Zu einem der ersten Gruppenempfehlungssystemen gehört *MusicFX* [McCarthy und Anagnost, 1998], welches 1998 vorgestellt wurde und zur Auswahl der passenden Hintergrundmusik für eine Gruppe in einem Fitnesscenter diente. Dabei wurden die Vorlieben der Mitglieder aufgenommen, das System erstellte durch Aggregation ein Gruppenprofil und wählte einen der empfohlenen Musikkanäle aus. Zusätzlich wurde mittels Zufallsprinzip für eine Variierung der Kanäle gesorgt.

Ein weiterer Musik-Gruppenempfehlungsdienst ist *Flytrap* [Crossen, Budzik und Hammond, 2002]. Basierend auf der Musik, die die Benutzer auf ihren Computern anhören, produziert Flytrap automatisch eine Musikauswahl die alle anwesenden Personen eines Raumes zufrieden stellt. Das System identifiziert dabei die Anwesenheit automatisch und berührungslos anhand der elektromagnetischen Ausweise. Danach werden die Empfehlungen durch Vereinigung der Benutzervorlieben berechnet.

Einen anderen Weg als die bisher erwähnten Gruppenempfehlungssysteme für Musik in einer gemeinsamen Umgebung schlägt *Adaptive Radio* [Chao, Balthrop und Forrest, 2005] ein. Das Empfehlungssystem erkennt dabei was die Benutzer nicht hören wollen und berechnet durch Aggregation der negativen Präferenzen eine Musikempfehlung die von allen als akzeptabel eingestuft werden kann.

Das Musik-Gruppenempfehlungssystem *jMusicGroupRecommender* [Christensen und Schiaffino, 2011] baut auf einer Applikation zur Verwaltung und Wiedergabe von Musikdateien auf. Das originale System lieferte dabei bereits Empfehlungen für individuelle Benutzer mittels kollaborativen Filter-Methoden [Goldberg, Nichols, Oki und Terry, 1992; Lee, Cho und Kim, 2010]. Auf Basis der Benutzerpräferenzen und der Interaktion mit dem System werden Benutzerprofile erstellt, die dann zur Abschätzung passender Gruppenempfehlungen herangezogen werden. Durch eine zusätzliche Erklärung über das Zustandekommen der Empfehlungen, unterstützt das System die Benutzer dabei Einigkeit zu erzielen.

2.1.2 Film und Fernsehen

Im Bereich der Filmempfehlungen stellt *PolyLens* [O'Connor, Cosley, Konstan und Riedl, 2001] eine Erweiterung des *MovieLens* Empfehlungssystems um die Möglichkeit von Gruppenempfehlungen dar. Bei *MovieLens* handelt es sich um eine weitverbreitete Filmdatenbank, die der Entwicklungsgruppe *GroupLens* der Universität von Minnesota - USA ins Leben gerufen wurde. Neben der Klassifizierung der Filme anhand ihrer Genre besteht die *MovieLens* Datenbank aus den spezifischen Interessen und den individuellen Bewertungen der Benutzer sowie deren persönlichen und demografischen Merkmalen, wie Alter oder Geschlecht. Mit diesen gesammelten Daten und unter Zuhilfenahme einer kollaborativen Filtermethode wird eine Liste mit individuellen Filmempfehlungen berechnet. Diese Empfehlungsliste der einzelnen Mitglieder einer Gruppe werden in der *PolyLens*-Erweiterung mittels einer Aggregationsstrategie vereinigt, wobei bereits bewertete und somit gesehene Filme gemieden werden. Die Benutzer treten jedoch Teile ihrer Privatsphäre ab, da bei der Präsentation der Gruppenempfehlungen die individuellen Empfehlungen der Benutzer von allen einsehbar sind.

Ein weiteres Gruppenempfehlungssystem aus dem Bereich der Videotechnik ist das *Family Interactive TV system* (FIT) [Goren-Bar und Glinansky, 2002], welches als personalisierter Wegweiser des Fernsehprogramms entsprechend der Präferenzen der Fernsehzuschauer fungiert. Dabei werden die Informationen zu den Interessen der Benutzer implizit durch das Zuschauerverhalten eruiert.

Ähnlich dazu stellt *Yu's TV Recommender* [Yu, Zhou, Hao und Gu, 2006] ein Gruppenempfehlungssystem für den Bereich der Fernsehprogrammempfehlung dar. Dabei werden individuelle Präferenzen der Benutzer über die Bereiche wie Genre, Schauspieler oder andere Schlüssel aufgenommen und Benutzerprofile erstellt. Diese werden dann durch eine Variation der *Average*-Strategie mittels Gesamtabstandsminimierung (engl. total distance minimisation) zu einem Gruppenprofil vereint.

Das Gruppenempfehlungssystem für Filme, *jMoviesGroupRecommender* [Christensen und Schiaffino, 2011] stammt von den Entwicklern von *jMusicGroupRecommender*, das ein zuvor beschriebenes Empfehlungssystem für Musik darstellt. Auch dieses System bietet individuelle Empfehlungen für seine Benutzer an. Über explizites Feedback der Benutzer bezieht das

System seine Informationen und berechnet anhand dieser Evaluierung die Korrelationen zwischen den Filmen. Anhand dieser individuellen Empfehlungen und durch Einbeziehung von Bewertungen schätzt das System passende Gruppenempfehlungen ab. Dabei stehen unterschiedliche Arten von Algorithmen zur Auswahl. Ebenfalls liefert das System eine Erklärung über das Zustandekommen der Empfehlungsliste.

2.1.3 Essen, Reisen und Tourismus

Der *Pocket Restaurant Finder* [McCarthy, 2002] liefert Empfehlungen zu Restaurants für die gesamte Gruppe auf mobilen Geräten. Dafür werden zuerst von jedem Benutzer seine kulinarischen Präferenzen wie Art der Küche, Preiskategorie oder Entfernung anhand einer 5-Sterne-Skala aufgenommen. Mit Hilfe eines *Average*-Algorithmus werden die Präferenzen der Personen einer Gruppe zusammengelegt und dann anhand des momentanen Standorts eine Liste mit passenden Restaurants berechnet. Ein inhaltsbasierter Algorithmus sortiert dabei die Liste nach der Attraktivität für den Benutzer.

Intrigue (INteractive TouRist Information GUIdE) [Ardissono, Goy, Petrone, Segnan und Torasso, 2003] empfiehlt anhand von Präferenzen Touristenattraktionen einer heterogenen Gruppe und stellt mögliche Terminplanungen auf. Bei der Empfehlung wird mittels Bildung von Untergruppen und der Einschätzung ihrer Relevanz (z.B. werden Kinder und Menschen mit besonderen Bedürfnissen wichtiger bewertet) eine Aggregation der individuellen Vorlieben der Personen erstellt. Das Ergebnis wird dann inklusive einer Erklärung der Gruppe präsentiert.

Ein weiteres interaktives System ist das *Travel Decision Forum* [Jameson, 2004], welches eine Gruppe bei der Planung eines gemeinsamen Urlaubs unterstützt. Ein Vermittler führt dabei die Interaktionen zwischen den Benutzern und hilft ihnen sich auf gemeinsame Anforderungen zu einigen. Das System berücksichtigt bei seinen Empfehlungen sogar die Präferenzen der Personen betreffend der Ausstattung der Hotels oder der Sehenswürdigkeiten in deren Umgebung.

Ein weiteres Empfehlungssystem für die Planung eines gemeinsamen Urlaubs ist das *Collaborative Advisory Travel System* (CATS) [McCarthy, Salamó, Coyle, McGinty, Smyth

und Nixon, 2006], welches rund um das interaktive PC Interface von Circle Twelve Inc., dem DiamonTouch Tisch entwickelt wurde und auf dem die simultane Zusammenarbeit von bis zu vier Benutzern möglich ist. Die bei der Reise einzuhaltenden individuellen Kriterien werden vom System aufgenommen und zu Profilen der Benutzer vereint. Danach konstruiert das System ein Präferenzmodell der Gruppe, welches die Qualität der unterschiedlichen Reiseangebote anhand dieser Restriktionen berechnet.

2.1.4 Andere und domänenunabhängige Gruppenempfehlungssysteme

Einer gänzlich anderen Art der Gruppenentscheidungshilfe widmet sich die Internet-Browser Erweiterung *Let's Browse* [Lieberman, Dyke und Vivacqua, 1999]. Das System unterstützt eine Gruppe, indem es Vorschläge über Seiten mit möglichen gemeinsamen Interessen unterbreitet. Die dazu nötigen Informationen von jedem individuellen Nutzer sowie der Gruppe bekommt das System durch die Analyse der Wörter der besuchten Webseiten. Durch eine einfache lineare Kombination der Benutzerprofile berechnet das System eine Gruppenempfehlung einer Seite, die am besten in dem kombinierten Profil abgeschnitten hat. Zusätzlich bietet *Let's Browse* eine automatische Anwesenheitserkennung der Benutzer sowie eine Erklärung zu den Gruppenempfehlungen.

Alle bisher erwähnten Gruppenempfehlungssysteme beschränken ihre Empfehlungen auf eine Domäne, die nächsten vorgestellten Systeme sind jedoch domänenunabhängig. In einem Artikel von Garcia, Pajares, Sebastia und Onaindia [2012] wird ein Gruppenempfehlungssystem beschrieben, das in der Lage ist, für unterschiedliche Domänen eine Gruppenempfehlung abzugeben. Das Gruppenpräferenz-Modell wird durch Aggregation der Modelle der individuellen Benutzer erstellt, wobei in dem Artikel neben den bereits von anderen Gruppenempfehlungssystemen bekannten *Average* und *Average without Misery* Algorithmen, auch noch *Incremental Intersection* und *Incremental Collaborative Intersection* gegenübergestellt werden. Die individuellen Präferenzen werden mittels einer hybriden Funktion, bestehend aus demografischen, kollaborativen, inhaltsbasierten Daten sowie einer Filterung der allgemeinen Vorlieben, berechnet. Das Gruppenempfehlungssystem eruiert dann die Empfehlungen, die am besten diese Präferenzen erfüllen.

Ein weiteres domänenunabhängiges Gruppenempfehlungssystem ist *WeDecide* [Stettinger, Ninaus, Jeran, Reinfrank und Reiterer, 2013], das dieser Arbeit als Grundgerüst dient.



Abbildung 2.1: Hauptseite der WeDecide Web-Version

Das System bietet die Möglichkeit individuelle Gruppenentscheidungsaufgaben zu modellieren und leitet die Benutzer durch den Entscheidungsprozess. Zusätzlich bietet das System unterschiedliche Möglichkeiten zur Kommunikation in der Gruppe an. Bei der Modellierung stehen folgende fünf verschiedene Arten von Gruppenentscheidungsaufgaben zur Auswahl bereit [Stettinger, 2013]:

- **„Antworten finden“** - diese Art von Gruppenentscheidungsaufgaben unterstützt eine Gruppe beim Finden von gemeinsamen Antworten auf Fragen, wobei die Attribute im Vordergrund stehen. Ein Beispielszenario ist die gemeinsame Konfiguration eines Computers.
- **„Wähle eine Alternative“** - den Teilnehmern werden mehrere Alternativen zur Auswahl vorgelegt. Dabei wird auf Attribute verzichtet und es findet keine Vorselektion statt, wodurch sich dieses Szenario am besten für eine Aufgabenstellung mit wenigen Alternativen eignet oder wenn eine Vorselektion mittels Attribute nicht möglich ist. Die Auswahl eines passenden Brettspiels für einen Spieleabend ist für diese Art ein Beispielszenario.

- **„Suche & wähle eine Alternative“** - dieses Szenario stellt eine Kombination aus den beiden zuvor erwähnten Szenarien dar. Die Gruppenentscheidungsaufgabe eruiert anhand von Fragen Attribute, die dann die Liste mit den Alternativen einschränkt und so eine Auswahl durch die Teilnehmer erleichtert. Ein Beispielszenario ist ein Konfigurator für den Kauf eines Familienautos, bei dem die Benutzer die gewünschten Eigenschaften angeben und dann eine Liste mit passenden Autos angezeigt bekommen.
- **„Einen Termin finden“** - der Suche nach einem passenden gemeinsamen Termin widmet sich dieses Szenario. Dabei gibt jeder Teilnehmer seine individuellen zeitlichen Rahmen bekannt und das System liefert passende Empfehlungen für die Gruppe. Das finden eines geeigneten Termins für ein Teammeeting ist ein mögliches Beispielszenario.
- **„Anforderungen priorisieren“** - spezifische Anforderungen werden in diesem Szenario durch die Teilnahme der Benutzer nach ihrer Relevanz gestaffelt. Ein Beispielszenario ist die Priorisierung von Anforderungen einer Softwareentwicklung.

Für die Aufgaben stehen unterschiedliche Algorithmen zur Empfehlungsberechnung zur Verfügung, aus denen der Benutzer eine Auswahl treffen kann. Die Abbildung 2.1 zeigt die Hauptseite des *WeDecide* Systems, das bis zu dieser Arbeit und dem damit verbundenen Projekt nur als Web-Version verfügbar war.

2.2 Klassifikation von Gruppenempfehlungssystemen

Die Funktionen, die Gruppenempfehlungssysteme auszeichnen können, wurden in dem Artikel über *Preference elicitation techniques for group recommender systems* in unabhängige Bereiche unterteilt [Garcia, Pajares, Sebastia und Onaindia, 2012]. Eine ähnliche Klassifizierung findet sich in *Group Recommender Systems: Combining individual models* [Masthoff, 2011]. Nachfolgend werden diese Bereiche erklärt und mit den zuvor erwähnten Gruppenempfehlungssystemen in Verbindung gebracht.

In der Informationsgewinnung bei Empfehlungssystemen kann grundsätzlich zwischen inhaltsbasiertem und kollaborativem Filtern unterschieden werden. Jedoch sollte zumindest erwähnt werden, dass es auch Arbeiten mit demografischer [Pazzani, 1999; Vozalis und Margaritis, 2007] oder wissenbasierter [Felfernig und Burke, 2008; Towle und Quinn, 2000] Vorgehensweise gibt. Beim inhaltsbasiertem Filtern werden Ähnlichkeiten von Elementen anhand deren Metadaten definiert. Basierend auf diesen Metadaten und den Bewertungen eines Benutzers wird dessen Profil erstellt. Die Prognosen über möglicherweise interessante Elemente, die der Benutzer noch nicht bewertet hat, werden anhand der Präferenzen dieses Profils erstellt. *MusicFX*, *Intrigue* oder *CATS* sind Systeme die auf inhaltsbasiertes Filtern setzen. Die Methode des kollaborativen Filtern benötigt keine explizite Eingabe der Metadaten von Elementen. Auf Basis des implizit beobachteten Verhaltens der Benutzer werden Ähnlichkeitsmatrizen gebildet. Dabei wird zwischen benutzerbasiertem und elementbasiertem Ansatz unterschieden. Der benutzerbasierte Ansatz bildet die Ähnlichkeitsmatrix zwischen den Benutzern und liefert Empfehlungen anhand der statistischen Nachbarn des Benutzers. Beim elementbasierten Ansatz wird die Ähnlichkeitsmatrix zwischen den Elementen gebildet, wobei Elemente als ähnlich bezeichnet werden, wenn Benutzer beide Elemente positiv bewertet haben. *PolyLens* und *Let's Browse* gehören zu den Gruppenempfehlungssystemen, die auf die Methode des kollaborativen Filtern setzt. Weiters gibt es auch noch hybride Methoden, die aus einer Kombination aus der genannten Methoden entstehen [Pazzani, 1999].

Bei der Interaktion der Gruppe mit dem System kann zwischen passivem und aktivem Verhalten unterschieden werden. Bei aktiver Teilnahme wird das finale Ergebnis erst durch eine abschließende Evaluierung der vom System erstellten Gruppenempfehlungen durch die Mitglieder einer Gruppe beschlossen. Zu diesen Systemen gehört das *Travel Decision Forum* und *CATS*. Bei passiver Teilnahme spricht das System eine Empfehlung aus, ohne das eine

weitere Interaktion vorgesehen ist, hier können *Intrigue* und *PolyLens* als Beispiele dienen.

Eine weitere Unterscheidung in ihrer Funktion ergibt sich durch domänenspezifische oder allgemeine Gruppenempfehlungssysteme, bei denen das System nicht nur auf eine Domäne beschränkt ist. Die meisten der vorgenannten Gruppenempfehlungssysteme sind auf eine Domäne spezialisiert, wie *FlyTrap* für den Musikbereich, *jMoviesGroupRecommender* für die Filmdomäne oder der *Pocket Restaurant Finder* der Empfehlungen zu Restaurants abgibt. Es gibt nur wenige Gruppenempfehlungssysteme die domänenunabhängig sind. Zu diesen gehört das Gruppenempfehlungssystem aus dem Artikel von [Garcia, Pajares, Sebastia und Onaindia, 2012] und selbiges, auf das diese Arbeit aufbaut: *WeDecide* [Stettinger, Ninaus, Jeran, Reinfrank und Reiterer, 2013].

Eine weitere Dimension ist die Menge der empfohlenen Produkte oder Inhalte. Bei manchem System genügt eine treffende Empfehlung, wie *MusicFX* oder *CATS*, da es ausreicht wenn ein passender Radiosender gespielt oder eine Urlaubsreise richtig geplant wird. Jedoch gibt es auch Aufgaben bei denen den Mitgliedern einer Gruppe eine Liste mit Empfehlungen, die möglicherweise sogar absteigend nach ihrer Relevanz sortiert sind, übergeben wird. Zu diesen gehören unter anderem *Intrigue*, denn eine Gruppe wird für normal nicht nur eine Touristenattraktion bewundern wollen oder *jMusicGroupRecommender*, da ein Musikstück für gewöhnlich nicht sehr lange ist und daher eine Abspielliste von Vorteil wäre.

Die Gruppengröße stellt ebenso einen relevanten Punkt dar. Dabei wird zwischen Systemen unterschieden, die nur mit einer geringen Anzahl an Mitgliedern zurechtkommt und solchen, bei denen die Gruppengröße keine Einwirkung auf das System hat. *MusicFX* oder *Intrigue* kommen mit einer sehr großen Anzahl an Mitgliedern klar, wogegen *PolyLens* oder *Let's Browse* nur Gruppen mit einer geringen Anzahl von Personen akzeptieren, um die Qualität der Empfehlung hoch zu halten.

Die Systeme beschreiten unterschiedliche Wege, um zu ihren Gruppenempfehlungen zu gelangen. Gruppenempfehlungssysteme wie *Intrigue* und *PolyLens* erstellen im ersten Schritt anhand der Benutzerpräferenzen die Empfehlungen für jede individuelle Person der Gruppe. Im zweiten Schritt werden diese Empfehlungen durch einen Algorithmus zu Gruppenempfehlungen vereinigt. Andere Systeme fassen die individuellen Präferenzen zu einem Gruppenmodell zusammen, dies kann explizit bewerkstelligt werden, indem die Benutzer zur Präferenzabgabe ein gemeinsames Gruppenkonto verwenden, oder implizit, durch Aggrega-

tion der individuellen Benutzerprofile. *MusicFX* generiert ein Gruppenprofil anhand einer Strategie, bei der nur die individuellen Bewertungen, die eine bestimmte Grenze übersteigen, zur Verwendung kommen. Beim *Travel Decision Form* kommt ein interaktives Verhandlungssystem zum Einsatz, bei dem ein Mediator-Agent die Aufgabe übernimmt, Werte für die Attribute des Gruppenmodells vorzuschlagen und für jedes Gruppenmitglied ein Agent eingesetzt wird, der dessen Interessen bei den Verhandlungen vertritt.

Alle diese Systeme verwenden unterschiedlichste Aggregationsmethoden um möglichst gute Empfehlungen abzuliefern, diese werden im nächsten Kapitel genauer behandelt.

2.3 Aggregationsmethoden

Die Aggregationsmethode ist eine mathematischen Funktion zur Vereinigung der individuellen Empfehlungen zu einer Gruppenempfehlung oder der Zusammenfassung der Präferenzen der individuellen Gruppenmitglieder zu einer Gruppenpräferenz. Die Wahl der Methode kann ausschlaggebend für die Qualität der Gruppenempfehlung sein.

Pessemier, Dooms und Martens [2013] hat sich dem Vergleich und der Evaluierung der Algorithmen durch die vier Aspekte angenommen: Genauigkeit (engl. accuracy), Diversität (engl. diversity), Abdeckung (engl. coverage) und Serendipität (engl. serendipity). Dabei wurden anhand der *MovieLens*-Datenbank fünf Algorithmen (*Average*, *Average without misery*, *One user choice*, *Least misery* und *Most pleasure*), die bereits in Masthoff [2004] vorgestellt wurden, und Kombinationen zwischen ihnen gegenübergestellt. Außerdem wurde die Gruppengröße, die bereits im Kapitel 2.2 zur Klassifikation von Gruppenempfehlungssystemen besprochen wurde, in den Experimenten sowie der Einfluss von Gemeinsamkeiten der Gruppenmitglieder auf die Qualität der Empfehlung berücksichtigt.

Ein wichtiger Punkt bei den Ergebnissen der Studie war, dass es keine allgemeine beste Lösung für den Empfehlungsalgorithmus und die Gruppierungsstrategie gibt. Um eine Optimierung der Gruppenempfehlung auf die gewünschten Aspekte zu erlangen, sollte der Empfehlungsalgorithmus und die Gruppierungsstrategie gemeinsam ausgesucht werden. Wird zum Beispiel auf die Aspekte Diversität und Abdeckung besonderer Wert gelegt, sollte laut der Studie ein elementbasierter kollaborativer Filter Algorithmus (engl. item-based collaborative filtering algorithm) oder ein hybrider Algorithmus mit der Aggregation der individuellen Empfehlungen eingesetzt werden.

2.4 Plattformübergreifende Entwicklung

Bei der Entwicklung von mobilen Applikationen liegt der Fokus auf der Präsentation und der Benutzerfreundlichkeit, wobei gute Geschwindigkeit, Leistung und Funktionalität der Applikation als selbstverständlich angesehen werden. Native Applikationen haben dabei einen klaren Vorteil, da sie mit der Programmiersprache des Systems und ohne zusätzlicher Kommunikationsebene arbeiten sowie mit der Hardware der Geräte effizienter interagieren können. Soll eine Applikation jedoch auf mehreren Betriebssystemen zum Einsatz kommen, liegt der Nachteil in der dafür notwendigen Entwicklungszeit für jede dieser Plattformen. Daher stellt neben den Gruppenempfehlungssystemen, die plattformübergreifende Entwicklung der mobilen Applikation einen Kernpunkt dieser Arbeit dar. Bei der sogenannte hybriden Entwicklung wird auf einer gemeinsamen Basis und ohne plattformspezifischem Wissen eine Applikation entwickelt, die dann in eine native Applikation der relevanten mobilen Betriebssystemen umgewandelt wird. Abbildung 2.2 dient der Verdeutlichung dieses Verfahrens.

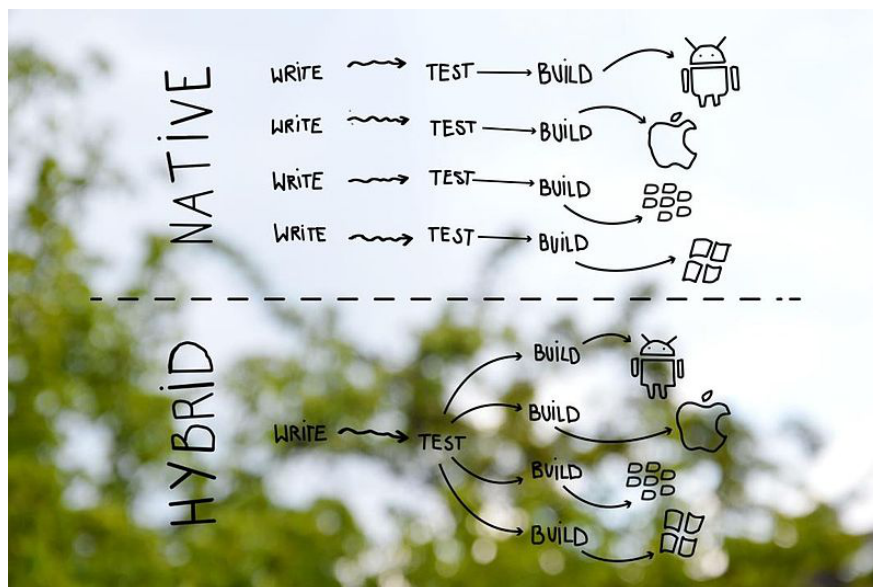


Abbildung 2.2: Beschreibung von Hybrid Apps [Hybrid Apps, 2014]

Es gibt bereits einige Arbeiten, die sich mit der Gegenüberstellung von nativen, web und hybriden Applikationsentwicklungen beschäftigen. Heitkötter, Hanschke und Majchrzak [2012] stellen umfassende Kriterien für die Bewertung der unterschiedlichen Ansätze auf und

führen anhand dieser eine Evaluierung durch. Die plattformübergreifende Entwicklung kann neben kürzerer Entwicklungszeit und geringerem Budget bei der Entwicklung für multiple Plattformen, auch beim Einsatz der Entwicklung für nur eine Plattform die beste Lösung darstellen.

Gagern [2013] untersucht die Vor- und Nachteilen der verschiedenen Entwicklungsansätzen von Native, Web- oder Hybrid-Applikationen. Die hybride Entwicklung kombiniert die Vorteile von Web-Applikationen und Native Entwicklungsansätzen, jedoch muss mit Performance-Einbußen gerechnet werden. Es wird dabei festgehalten, dass kein perfekter Ansatz vorhanden ist, die Entwickler jeden Einzelfall abwägen und für sich die beste Lösung in Sachen Features der Applikation, Zielgruppe, Budget und Entwicklungszeit wählen sollten.

3 Systemarchitektur

Die Architektur von *WeDecideMobile* entspricht einem Server-Client-Modell, bei dem Server und Client über eine Programmierschnittstelle miteinander kommunizieren. Dabei kommt das Model View Controller (MVC) Entwurfsmuster [Leff und Rayfield, 2001] zum Einsatz. Abbildung 3.1 zeigt die Architektur des Systems.

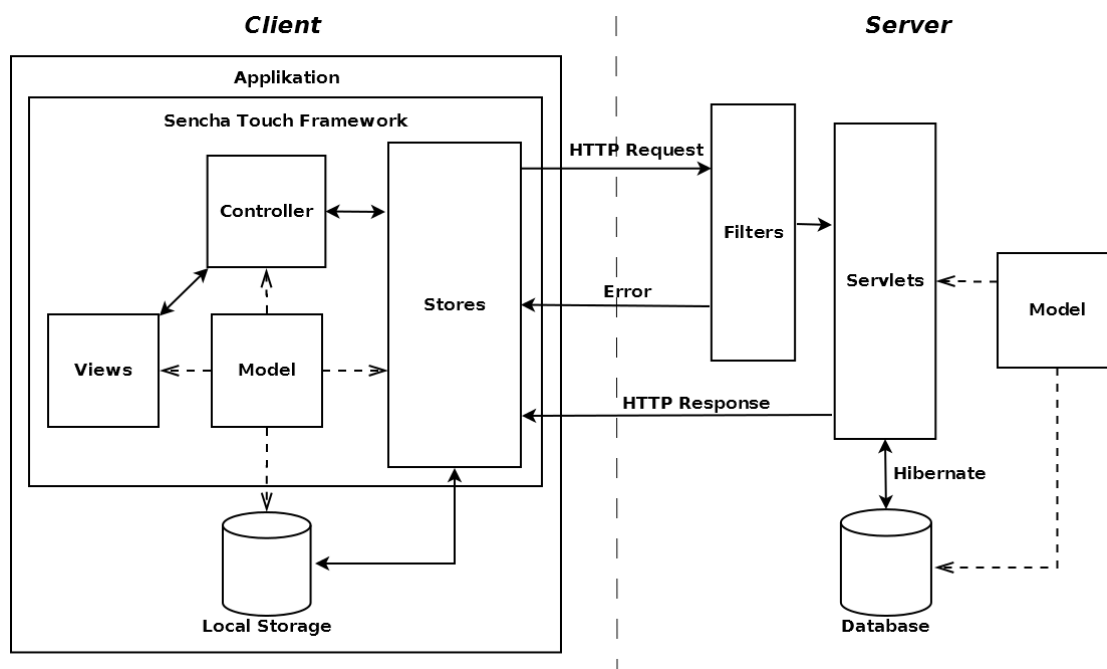


Abbildung 3.1: Architektur des Systems *WeDecideMobile*

Das Kapitel 3.1 behandelt den Aufbau des Systems, erörtert das MVC Entwurfsmuster und liefert eine Übersicht der Schnittstelle. Die verwendeten Technologien werden in Kapitel 3.2 beschrieben. Das letzte Kapitel 3.3 beschäftigt sich mit der Entwicklung des Projekts.

3.1 Aufbau des Systems

Der Server besteht aus dem Open Source Webserver Apache Tomcat [Apache Tomcat, 2013] und dem Datenbankverwaltungssystem MySQL [MySQL, 2013]. Zur Bearbeitung und Beantwortung von Anfragen (engl. Requests) des Clients, kommen auf dem Webserver Java Servlets zum Einsatz, die dabei mit der Datenbank via Hibernate kommunizieren und die Daten, wie in MVC-Architekturen üblich, mittels Datenmodellen austauschen.

Da der Webserver nur zur Datenbereitstellung dient und für keine visuelle Darstellung zuständig ist, wird auf eine Präsentation mittels Java Server Pages (JSP) verzichtet. An dessen Stelle tritt zur Datenübertragung zwischen Server und Client eine Programmierschnittstelle, die Daten mit dem Datenformat JavaScript Object Notation (JSON) an den Client sendet.

Der Client entspricht einer mobilen Applikation, die sich des JavaScript Frameworks *Sencha Touch* bedient. Dieses Framework basiert dabei komplett auf Web Standards wie Javascript, Hypertext Markup Language 5 (HTML5) und Cascading Style Sheets 3 (CSS3) und bietet den Entwicklern die Möglichkeit HTML5 basierende mobile Applikationen zu erstellen, die durch die Cross-Plattform Entwicklung auf den gängigsten mobilen Betriebssystemen funktionieren. Die Applikation wurde mittels testgetriebener Entwicklung (engl. *Test Driven Development*) unter Verwendung von *Jasmine* erstellt.

3.1.1 Model View Controller

Model View Controller (MVC) (dt. Modell-Präsentation-Steuerung) ist ein Entwurfsmuster zur Trennung von Geschäftslogik und User Interface [Freeman und Freeman, 2006], wie in Abbildung 3.2 dargestellt.

Diese saubere Trennung ermöglicht einen flexiblen Programmentwurf, wodurch die spätere Wartbarkeit des Programms und die Wiederverwendbarkeit einzelner Komponenten erleichtert werden. Die Entwicklung wird dabei in drei Bestandteile strukturiert:

- Das Datenmodell (engl. model) enthält die anzuzeigenden Daten sowie die Regeln, um diese Daten zu manipulieren und ist von der Steuerung und Präsentation unabhängig.

- Die Ansicht (engl. view) dient der Darstellung und Präsentation der Informationen aus dem Datenmodell, nimmt Interaktionen mit den Benutzern auf und gibt diese an die Steuerung weiter.
- Die Steuerung (engl. controller) dient der Kommunikation zwischen Datenmodell und Ansicht. Dabei verwaltet sie mehrere Ansichten, nimmt deren Interaktionen mit dem Benutzer entgegen, verarbeitet diese und führt die entsprechenden Aktionen aus. Weiters erstellt, aktualisiert oder löscht sie Daten in den Datenmodellen und gibt Veränderungen der Daten an die Ansicht weiter.

Das Konzept für Model View Controller wurde vom Unternehmen Xerox PARC 1978 für Benutzeroberflächen in Smalltalk entwickelt und wird dem Entwickler Trygve Reenskaug zugerechnet. Die erste offizielle Veröffentlichung fand erst August/September 1988 statt [Krasner und Pope, 1988]. [M. H. Reenskaug, 2003]

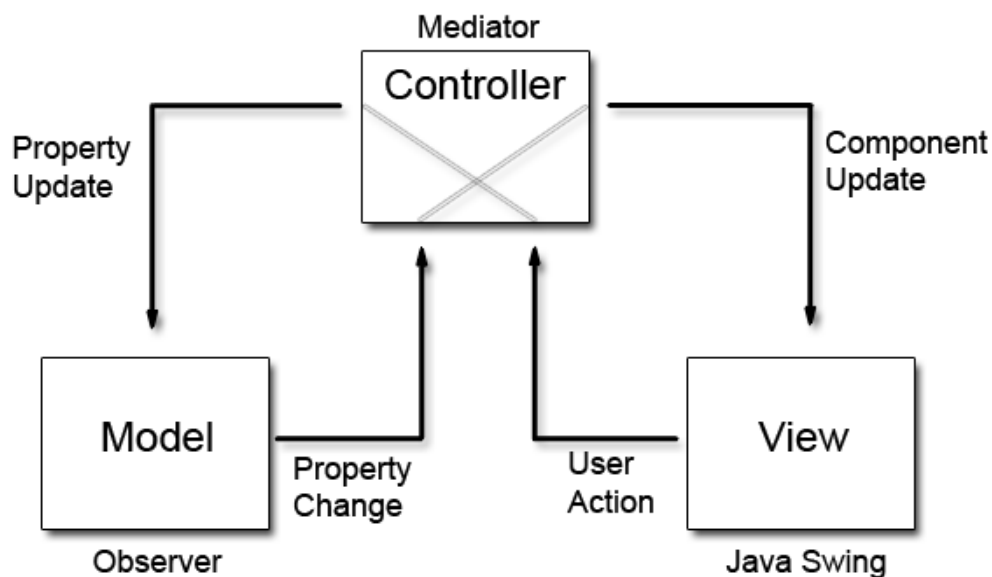


Abbildung 3.2: Model View Controller Entwurfsmuster [Simple Example of MVC, 2008]

3.1.2 Schnittstellenübersicht

Für das vorhandene *WeDecide*-System wurde eine Schnittstelle konzipiert, die die Funktionen der Web-Version widerspiegelt und Kommunikation mit dem Client ermöglicht. Daten werden dabei mit dem Datenformat JSON übertragen. Da der Client auf Web-Technologien setzt und somit zum Server eine unterschiedliche Domäne aufweist, tritt bei der Übertragung von Daten das Sicherheitskonzept zum Schutz vor Angriffen in modernen Browsern und Web-Anwendungen, die *Same-Origin-Policy (SOP)* [Same-origin policy, 2013] in Kraft, die clientseitigen Skriptsprachen wie Javascript, aber auch Cascading Style Sheets (CSS) untersagt auf Ressourcen einer anderen Domain zuzugreifen. Durch den Einsatz von JSONP wird diese Richtlinie umgangen und die Datenübertragung kann durchgeführt werden. Jedoch stehen dadurch nur mehr die *GET*-Methode zum Datenaustausch zur Verfügung, auf die *Post*-Methode muss leider verzichtet werden.

Die komplette Schnittstelle wird durch einen Filter geschützt, der mehrere Aufgaben erfüllt:

- Erste Aufgabe - kein Anfrage an die Schnittstelle darf ohne Variable für den Namen der Callback-Funktion gestellt werden, die in JSONP zur Einbettung der Daten dient.
- Zweite Aufgabe - der Filter deaktiviert Caching von Seiten des Servers.
- Dritte Aufgabe - der Parameter für die Sprache wird abgefangen und die Sprache im System eingestellt.
- Vierte Aufgabe - der Content-Type der Antwort wird vom Filter auf „application/javascript“ gestellt, da dies bei JSONP zur korrekten Verarbeitung notwendig ist.

Da sowohl der Server als auch der Client auf die MVC-Architektur setzen und Daten innerhalb der Systeme in Objekt-Form weitergegeben werden, war es notwendig eine Klasse zu entwickeln, die für das Umwandeln von Objekt zu JSON-Darstellung zuständig ist. Dabei wird rekursiv vorgegangen. Weiters wurde eine Klasse zum Error-Handling implementiert, die den Client über mögliche Fehler und deren Ursache unterrichtet.

3.2 Verwendete Technologien

3.2.1 Apache Tomcat



Apache Tomcat [Apache Tomcat, 2013] ist ein Webserver und Servlet Container, der von der Apache Software Foundation (ASF) als Open-Source-Projekt entwickelt wurde und die Spezifikationen von Java Servlets und Java Server Pages implementiert. Dadurch bietet der Apache Tomcat die Möglichkeit, in Java geschriebene Web-Anwendungen auf Servlet- und JSP-Basis auszuführen. Da der Webserver auf Java basiert, benötigt Java Runtime Enterprise

Environment 1.1 oder höher.

Die Entwicklung des Tomcat Projekts [The Tomcat Story, 2014] bei Sun Microsystems startete als Referenzimplementation der Java Servlet und Java Server Pages Spezifikation. Der Tomcat Sourcecode wurde von Sun Microsystems 1999 an die Apache Software Foundation gestiftet und seit 2005 ist Tomcat ein Top-Level Apache Projekt. Das erste Release von Apache war in Version 3.0, seitdem gab es weitere 12 große Veröffentlichungen. Apache Tomcat ist seit 17. Februar 2013 in Version 7.0.52 erhältlich.

3.2.2 Java Servlets



Java Servlets [Ullenboom, 2011] sind Java-Klassen, die die javax.servlet.Servlet Schnittstelle implementieren. Da in diesem Projekt alle Anfragen (engl. Requests) über das Hypertext Transfer Protocol (HTTP) Protokoll erfolgen und dieses die wichtigen Arbeitsmethoden enthält, erweitern die Java Servlets sogar HttpServlet.

Die Servlets nehmen Anfragen vom Client entgegen, verarbeiten diese und liefern eine Antwort (engl. Response). Diese Antworten werden dabei dynamisch erstellt.

Das Konzept der Servlets wurde 1994 im Zusammenhang mit der Entwicklung des ersten vollständig auf Java basierendem Webserver definiert. Die Servlets hatten jedoch den Nachteil der Koppelung von Visualisierung und Logik, die kurze Zeit danach durch die Technik der Java Server Pages (JSP) vermieden wurde. Die im Juli 1997 von der Firma Sun veröffentlichten Java-Servlet-Spezifikation mit der Version 1.0 wurde ab Version 2.3 unter der Java Community Process entwickelt. Seit Mai 2013 ist die Servlet-Spezifikation in der Version 3.1 vorhanden. Verschiedene lizenzfreie Webserver mit Servlet-Funktionalität erfüllen diese Spezifikation, unter anderem der bereits erwähnte Apache Tomcat. [Servlet, 2014]

3.2.3 Hibernate



Hibernate [Hibernate, 2013] ist ein Framework für objektrelationale Abbildungen in Java. Es bietet die Möglichkeit Objekte die aus Domain Modellen erstellt wurden in einer relationalen Datenbank zu speichern und sie wieder zu laden, dabei werden Beziehungen zwischen den Objekten auf entsprechende Datenbank-Relationen abgebildet. Außerdem übernimmt Hibernate das Generieren von SQL-Statements, befreit den Entwickler somit vom manuellen Umgang mit *ResultSet* und ermöglicht dadurch das leichte Austauschen des Datenbankverwaltungssystems. Des Weiteren bietet Hibernate Search eine einfache Möglichkeit von Volltextsuche mittels Apache Lucene[Apache Lucene, 2013].

Das Hibernate-Framework wurde 2001 von Gavin King und seinem Team beim Unternehmen Cirrus Technologie als Fehlerbehebung für ein bekanntes Problem der *Entity Beans* bei der *J2EE* (Java 2 Enterprise Edition) Entwicklung entworfen. 2003 wurde Hibernate in Version 2 mit signifikanten Verbesserungen zu seinem Vorgänger veröffentlicht. Das Unternehmen JBoss Inc. stellte einige der wichtigsten Hibernate-Entwickler ein, darunter auch Gavin King, um die weitere Entwicklung von Hibernate voranzutreiben [Gavin King, 2004]. In der Mitte von 2006 wurde JBoss Inc. von Red Hat, einem sehr bekannten Linux-Distributor, übernommen. Die letzte Veröffentlichung mit der Version 4.1.9 war im Dezember 2012, aber die Weiterentwicklung von Hibernate zur Version 5 ist bereits im vollen Gange. [Hibernate ORM, 2012]

3.2.4 MySQL



MySQL [MySQL, 2013] ist ein relationales Datenbankverwaltungssystem und als Open-Source-Software unter der GNU General Public Lizenz erhältlich. Seit Juli 2013 [DB-Engines Ranking, 2014] ist MySQL das zweitpopulärste Datenbankverwaltungssystem der Welt.

MySQL wurde vom schwedischen Unternehmen MySQL AB entwickelt und setzt sich aus dem Namen der Tochter *My* des Mitbegründers Michael Widenius und SQL, der Abkürzung für Structured Query Language, zusammen. SQL ist eine relationale Datenbanksprache und setzt sich aus einer Datenbeschreibungssprache, mit der Tabellen für Daten in der Datenbank angelegt und gewartet werden, und einer Datenmanipulationssprache, mit der die Datensätze in den Tabellen angelegt, verändert oder gelöscht werden können, zusammen. Seit Jänner 2010 gehört das Unternehmen der Oracle Corporation und wird von dieser betreut.

3.2.5 JavaScript



Die Skriptsprache JavaScript ist eine vielseitige und flexible Programmiersprache, mit der sich komplexe Anwendungen entwickeln lassen. Ursprünglich wurde JavaScript entwickelt, um dynamisch Einfluss auf statische HTML-Seiten in Web-Browsern zu nehmen, wie deren Funktionalität zu steuern, Inhalte des HTML-Dokuments zu verändern, nachzuladen oder zu generieren sowie Benutzerinteraktionen auszuwerten. Dabei liest ein Interpreter, der in allen modernen Web-Browsern integriert ist, den Programmcode ein und führt die darin beschriebenen Aktionen aus. Die Ausführung geschieht dabei clientseitig und entlastet somit den Server. Heutzutage kommen JavaScript-Programme auch außerhalb der Web-Browser zum Einsatz, wie im Server-Bereich oder auf Microcontrollern.

JavaScript wurde ursprünglich 1995 von der Firma Netscape Communication Corp. für die neueste Version ihres Internet-Browsers Netscape Navigator unter dem Namen LiveScript entwickelt. Da sich der Syntax der Sprache an Java orientiert, wurde aus marketingtechnischen

Gründen die Programmiersprache in JavaScript umbenannt. Nachdem das Unternehmen Microsoft eine ähnliche Skriptsprache mit dem Namen JScript für ihren Internet Explorer veröffentlichte, brach der sogenannte erste Browserkrieg [15 Jahre WWW, 2008] zwischen den beiden Rivalen um die Vorherrschaft im Internet aus. Um ein zu weites Auseinanderentwickeln der beiden Sprachen zu verhindern, wurde 1997 von der Standardisierungsorganisation ECMA ein einheitlicher Grundstandard, dem ECMAScript (ECMA 262), veröffentlicht.

In den darauf folgenden Jahren wurde JavaScript ständig weiterentwickelt und ist derzeit in der Version 1.8.5 erhältlich, der im freien Firefox-Browser von Mozilla zum Einsatz kommt. Dem gegenüber steht die im Internet Explorer ab Version 9 zum Einsatz kommende JScript-Sprache mit der stabilen Version 9.0. [Koch, 2011; Wenz, 2007]

3.2.6 JSON und JSONP



JSON (JavaScript Object Notation) ist ein leichtgewichtiges Datenformat, das für Menschen einfach zu lesen und schreiben sowie für Maschinen leicht zu parsen und generieren ist. Dabei dient es dem Zwecke des Datenaustauschs zwischen unterschiedlichen Anwendungen und beruht auf der JavaScript Programmiersprache, von der es jedoch unabhängig ist. JSON-Parser gibt es in praktisch allen verbreiteten Programmiersprachen und es wird oft als ressourcensparender Ersatz für Extensible Markup Language (XML) eingesetzt. Im Dezember 1999 wurde JSON von Douglas Crockford, einem prominenten Verfechter von JavaScript und Mitarbeiter des Unternehmens Yahoo, spezifiziert. [Introducing JSON, 2014]

JSON *with Padding* (JSONP) wurde 2005 von Bob Ippolito vorgestellt und wird zur Übertragung von Daten im zuvor erörterten JSON-Datenformat verwendet, wobei die Datenzugriffe nicht auf den Server beschränkt sind, von dem die HTML-Seite geladen wurde. Somit kann mittels JSONP über Domänengrenzen hinweg auf alle Server und Web-Services zugegriffen werden, was für gewöhnlich am Sicherheitskonzept der *SOP* [Same-origin policy, 2013], die den Zugriff von clientseitigen Skriptsprachen wie Javascript auf Ressourcen einer anderen Domäne untersagt, scheitert. JSONP wird heutzutage von vielen Anwendungen des Web 2.0, wie jQuery und Web Services unterstützt. [JavaScript Object Notation, 2012]

3.2.7 Sencha Touch



Sencha Touch [Sencha Inc., 2013] wurde speziell zur Entwicklung von plattformübergreifenden mobilen Applikationen entwickelt und entstand durch die Vereinigung der bekannten JavaScript-Bibliothek Projekte von *Ext JS*, *JQTouch* und *Raphaël*. Für Applikationsentwickler steht Sencha Touch unter der freien kommerziellen oder der Open-Source Lizenz zur Verfügung [Sencha Touch Licensing, 2014].

Die erste Veröffentlichung von Sencha Touch war im Juli 2010 mit der Version 0.90 beta, wobei die erste stabile Version 1.0 im November 2010 herausgegeben wurde. Zu dieser Zeit hatte das Framework noch mit erheblichen Leistungsproblemen zu kämpfen und unterstützte lediglich die mobilen Betriebssysteme *Android* und *iOS*. Sencha Touch 2, das im März 2012 veröffentlicht wurde, verwendet eine Rendering-Engine die komplett auf CSS beruht und kompatibel zum Klassenladesystem von *Ext JS 4* ist, dadurch wurden die Leistungsprobleme größtenteils behoben. Seit November 2013 liegt Sencha Touch in Version 2.3.1 vor, dabei wurde *Adobe Cordova* inklusive Unterstützung für *Adobe PhoneGap Build* vollintegriert. Weiters unterstützt Sencha Touch bereits etliche mobile Plattformen, wie *iOS*, *Android*, *BlackBerry*, *Windows Phone* sowie verschiedene Browser, unter anderem *Android browser*, *Google Chrome for Android*, *Bada Mobile Browser*, *Kindle Fire Browser*, *Windows 8 IE10* and *Mobile Safari*.

Sencha Touch beruht dabei gänzlich auf JavaScript, weshalb sich die komplette Applikation auf nur eine HTML-Datei stützt, die den JavaScript-Code beinhaltet. Jegliche Daten- oder Ansichtänderung wird dabei mittels JavaScript im Document Object Model (DOM) vollzogen.

Das Framework bietet eine große Auswahl an User Interface (UI) Elementen, die von *iOS* inspiriert wurden: Angefangen von einfachen Formular-Elementen wie Textfeldern, Checkboxes, Radiofeldern und Selectfeldern inklusive grafisch ansprechenden Pickern, Slidern, Togglefeldern und vielen anderen. Weiters gibt es noch Buttons, Top- und Bottom-Toolbars sowie Tabbars, mit denen die Oberfläche der Applikation nahe an denen der nativen Applikationen herankommt.

Ebenso groß ist die Anzahl der verschiedenen Events, mit denen die Applikation vom Benutzer gesteuert wird. Daneben besteht die Möglichkeit CSS-Themes für die unterschiedlichen mobilen Betriebssystemen zu verwenden und es werden bei jeder Version mehr. Abbildung 3.3 zeigt die neuen Themes, die seit Version 2.3 auch zur Auswahl stehen.

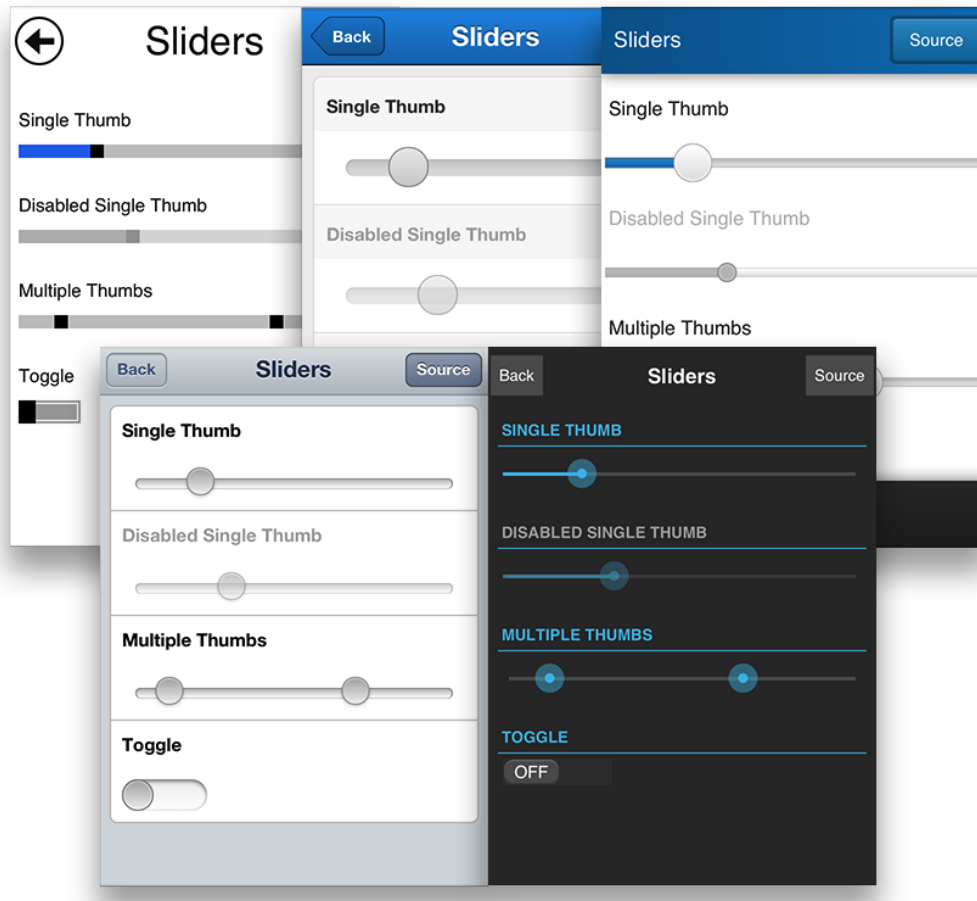


Abbildung 3.3: Neue Themes seit Sencha Touch 2.3 [Agrawal, 2013]

Das Styling wird dabei in Sencha Touch durch Syntactically Awesome Stylesheets (SASS), einer Metasprache zur Beschreibung von CSS, automatisch mittels *Compass* Interpreter erstellt. Neben den UI-Elementen stellt Sencha Touch einige Übergangsanimationen zur Verfügung, wodurch die Bedienung der Applikation weicher abläuft. Durch die Integration von *Adobe Cordova* werden eine Menge an nativen Features zur Verfügung gestellt. Nähere Informationen dazu liefert das nächsten Kapitel 3.2.8. [Kosmaczewski, 2013]

3.2.8 Apache Cordova



Apache Cordova [About Apache Cordova, 2014] ermöglicht den Zugriff auf native Funktionen, wie Kamera, Kompass oder Beschleunigungssensoren, aber auch Netzwerk, Medien oder Kontakte, von mobilen Geräten über JavaScript.

Durch die Kombination mit UI-Frameworks, wie dem JavaScript Framework Sencha Touch, ermöglicht Apache Cordova somit die Entwicklung von mobilen Applikationen, die nur auf Web-Technologien setzen und keine Notwendigkeit für nativen Code, wie Java oder Objective-C, benötigen. Dabei ist Apache Cordova in der aktuellen Version 2.0 auf den Plattformen iOS, Android, Blackberry, Windows Phone, Palm WebOS, Bada und Symbian verfügbar.

Apache Cordova, das ursprüngliche PhoneGap hieß, wurde 2011 von Adobe/Nitobi der Apache Software Foundation (ASF) gespendet [PhoneGap, Cordova, 2012], musste aber aus markenrechtlichen Gründen umbenannt werden. Im Oktober 2012 wurde Apache Cordova zu einem Top-Level-Projekt der Apache Software Foundation und soll auch weiterhin unter der freien und offenen Apache Lizenz 2.0 bleiben.

3.2.9 Jasmine



Jasmine ist ein Framework zur verhaltensgetriebenen Softwareentwicklung (engl. Behavior Driven Development), das in JavaScript implementiert wurde und auf allen Plattformen mit JavaScript ausführbar ist. Dabei ist Jasmine unabhängig von anderen JavaScript Frameworks, von Document Object Model (DOM) oder Web-

Browsern und eignet sich daher hervorragend zum Testen von Sencha Touch Applikationen. Die Testbeschreibungssprache ist einfach und klar gehalten und das Ausführen der Tests ist grundsätzlich performant, aber hängt dennoch von der JavaScript-Engine ab, auf die es angewendet wird. Jasmine ist derzeit in der Version 2.0 erhältlich und kann unter der MIT-Lizenz verwendet werden. [Jasmine, 2013; Kosmaczewski, 2012]

3.3 Entwicklung

Die Entwicklung der Applikation wurde mit Hilfe des *Sencha Architect*, einem Entwicklertool zur Erstellung von modularen und portablen grafischen Applikationen mit HTML5 und JavaScript [Sencha Architect Guide, 2014], durchgeführt. Der Entwicklungsprozess verwendet dabei den Ansatz der agilen Softwareentwicklung und setzt dabei auf verhaltensgetriebene Entwicklung (engl. Behavior Driven Development), die eine Erweiterung der testgetriebenen Entwicklung (engl. Test Driven Development) darstellt.

3.3.1 Testgetriebene Entwicklung

Testgetriebene Entwicklung (engl. Test Driven Development) ist ein Softwareentwicklungsprozess mit sehr kurzen Entwicklungszyklen, bei der die Tests vor dem wirklichen Produktionscode geschrieben werden. Dazu wurden von Martin und Coplien [2009] die drei Gesetze von testgetriebener Entwicklung definiert:

- Erstes Gesetz - Sie dürfen den Produktionscode erst schreiben, wenn Sie einen scheiternden Unit-Test geschrieben haben.
- Zweites Gesetz - Der Unit-Test darf nicht mehr Code enthalten, als für das Scheitern und ein korrektes Kompilieren des Tests erforderlich ist.
- Drittes Gesetz - Sie dürfen nur so viel Produktionscode schreiben, wie für das Bestehen des gegenwärtigen scheiternden Tests ausreicht.

Durch diese drei Gesetze werden die Entwickler dazu gebracht die Dauer des Zyklus auf unter 30 Sekunden zu minimieren, wobei Tests und Produktionscode gemeinsam geschrieben werden. Der Umfang der Tests kann dabei den Umfang des Produktionscodes einnehmen und den Entwickler vor Verwaltungsprobleme stellen, daher sollten die Tests immer schnell und sauber gehalten werden und der Qualität des Produktionscodes gleichen. Zu diesem Zweck wurden in Martin und Coplien [2009] fünf Eigenschaften aufgestellt, die zu stabilen und qualitativ hochwertigen Komponenten führen und zusammen die Abkürzung „**F.I.R.S.T.**“ bilden:

- **Fast** (dt. schnell) - Tests müssen schnell sein, sind sie es nicht, werden sie nicht regelmäßig ausgeführt.
- **Independent** (dt. unabhängig) - Tests sollen nicht voneinander abhängig sein und unabhängig voneinander ausgeführt werden können. Damit wird durch einem fehlgeschlagenen Test, eine Kaskade aus weiteren nicht bestandenen Tests verhindert, was die Diagnose erschweren würde.
- **Repeatable** (dt. wiederholbar) - Tests sollten unabhängig von der verwendeten Test-Umgebung wiederholbar sein, stets gleich funktionieren und die gleichen Ergebnisse in den unterschiedlichen Umgebungen liefern.
- **Self validating** (dt. selbst-validierend) - Test sollten stets einen booleschen Output (bestehen oder fehlschlagen) haben, wodurch das Vergleichen von Dateien oder das Studieren von Protokollen nicht notwendig ist.
- **Timely** (dt. zeitgerecht) - Tests sollten zum richtigen Zeitpunkt geschrieben werden, also kurz bevor der Produktionscode erstellt wird, der sie bestehen lässt.

Der größere zeitliche Aufwand der durch testgetriebenes Entwickeln anfällt, wird durch die entstehenden Vorzüge, der Qualität des Codes, die hohe Flexibilität in der Weiterentwicklung sowie die Austauschbarkeit und Wiederverwendbarkeit der Komponenten, belohnt [Bhat und Nagappan, 2006].

Die testgetriebene Entwicklung im Entwicklungsprozess von mobilen Applikationen ist schwer umzusetzen, da sich Tests für das Benutzerinterface nur schwierig implementieren lassen. Dafür eignen sich eigens entwickelte Frameworks, die jedoch bei jeder Abweichung oder Erweiterung der UI-Elemente ebenfalls angepasst und erweitert werden müssen. Auch können grafische Interaktionen oder Animationen nur schwer mittels Test beschrieben werden. Dafür wurde verhaltensgetriebene Entwicklung eingeführt, die im nächsten Kapitel behandelt wird.

3.3.2 Verhaltensgetriebene Entwicklung

Verhaltensgetriebene Entwicklung (engl. Behavior Driven Development (BDD)) ist eine Weiterentwicklung von testgetriebener Entwicklung von Dan North, bei der das testorientierte Vokabular der testgetriebenen Entwicklung gegen eine natürlichere Sprache zur Beschreibung von Aufgaben, Zielen und Ergebnissen ersetzt wird und erleichtert somit auch das Verständnis von Personen die keine Entwickler sind. Die gemeinsame Sprachbasis von Entwicklern, Projektmanagern und Kunden bietet die Voraussetzung für eine einfache und direkte Kommunikation. Die verhaltensgetriebene Entwicklung wurde 2006 erstmals in einem Artikel vom *Better Software* Magazin erwähnt [North, 2006]. Abbildung 3.4 zeigt ein Beispiel anhand von *Cucumber* [Cucumber, 2014]

```
1  # Language: de
2  Funktionalität: Division
3  Um dumme Fehler zu vermeiden
4  müssen Kassierer in der Lage sein einen Bruchteil zu berechnen
5
6  Szenario: Normale Zahlen
7  Gegeben sei ich habe 3 in den Taschenrechner eingegeben
8  Und ich habe 2 in den Taschenrechner eingegeben
9  Wenn ich divide drücke
10 Dann sollte das Ergebniss auf dem Bildschirm 1.5 sein
```

Abbildung 3.4: Beispiel für ein Verhalten in Cucumber [Cucumber Division Feature, 2014]

4 Grafische Benutzeroberfläche

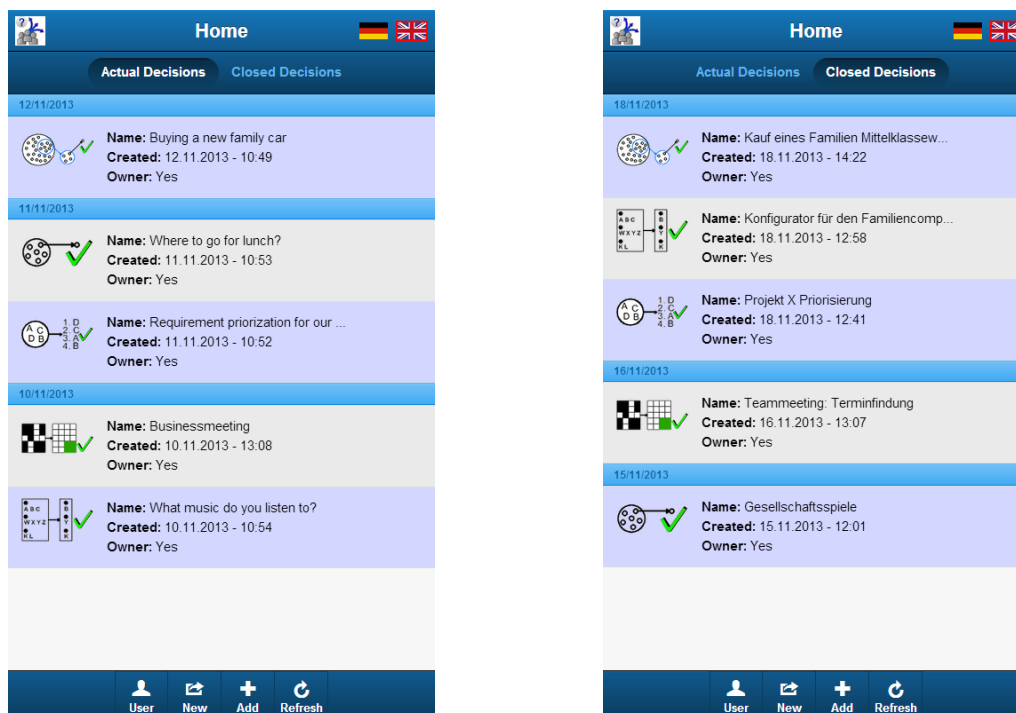
Dieses Kapitel beschäftigt sich mit der Implementation und dem Aufbau der grafischen Benutzeroberfläche. Die Aufnahme der Daten zur Kommunikation zwischen Client und Server geschieht dabei grundsätzlich über Formulare. Diese werden mittels Constraints (Bedingungen, die zwingend vom Wert einer Variable erfüllt werden müssen) direkt am Client validiert. Nachdem die Daten an den Server geschickt wurden, werden diese abermals validiert. Dieser Mehraufwand in der Überprüfung rechnet sich durch Sicherheit und den geringeren Kommunikationsaufwand mit dem Server.

Die folgenden Kapitel erläutern die unterschiedlichen Bereiche der grafischen Benutzeroberfläche von *WeDecideMobile*. Kapitel 4.1 erörtert die Hauptseite, die als Ausgangspunkt für alle Aktionen dient und die damit verbundene Verwaltung von bereits existierenden Gruppenentscheidungsaufgaben. Das nächste Kapitel 4.2 beschäftigt sich mit der Detailansicht einer Gruppenentscheidungsaufgabe und erörtert die verschiedenen Funktionen, die für die Teilnehmer bereit stehen. Das darauf folgende Kapitel 4.3 behandelt den Benutzerbereich und erklärt die unterschiedlichen Arten von Benutzern im System. Der Ablauf von Gruppenentscheidungsaufgaben, angefangen bei deren Modellierung bis zum finalen Ergebnis, wird in Kapitel 4.4 behandelt. Im Anschluss werden dann die fünf verschiedenen und unterstützen Szenarien der Gruppenentscheidungsaufgaben im Detail und unter Zuhilfenahme eines individuellen Beispiels je Szenario beschrieben.

Den Anfang machen dabei die drei *Choice Decision*-Szenarien, wobei „Antworten finden“ in Kapitel 4.5, „Wähle eine Alternative“ in Kapitel 4.6 und „Suche & wähle eine Alternative“ in Kapitel 4.7 im Detail erklärt werden. Danach wird in Kapitel 4.8 das „Einen Termin finden“ Szenario anhand eines Beispiels behandelt und zum Schluss kommt noch das „Anforderungen priorisieren“ Szenario in Kapitel 4.9. Die dabei zum Einsatz kommenden Abbildungen sind in englischer Sprache gehalten, da dies die native Sprache des Systems darstellt.

4.1 Hauptseite

Die Hauptseite, die in Abbildung 4.1 dargestellt ist, dient als Ausgangspunkt für alle Aktionen und zur Verwaltung bereits existierender Gruppenentscheidungsaufgaben. Diese werden der Übersicht halber in aktuelle und geschlossene Entscheidungen unterteilt. Die Listen sind dabei chronologisch sortiert und in Tage aufgeteilt. Sie beinhalten alle vorhandenen Gruppenentscheidungsaufgaben von dem registrierten Benutzer sowie allen vorhandenen anonymen Benutzern. Der Unterschied zwischen den Arten von Benutzern wird im Kapitel 4.3 genauer erörtert.



(a) Aktuelle Entscheidungen

(b) Geschlossene Entscheidungen

Abbildung 4.1: Hauptseite

Die einzelnen Listeneinträge bestehen aus einem Icon, welcher die Art der Gruppenentscheidungsaufgabe angibt, dem Titel und dem Datum der Erstellung, sowie der Information ob es sich bei dem Benutzer um den Ersteller der Aufgabe handelt. Dies ist von Bedeutung, da nur der Ersteller für die Administration der Gruppenentscheidungsaufgabe verantwortlich ist. Bei

Auswahl einer der Listeneinträge wird die Detailansicht, die im nächsten Kapitel genauer erklärt wird, der dabei ausgewählten Aufgabe angezeigt.

Weiters enthält die Hauptseite das Hauptmenü, wie in Abbildung 4.2 dargestellt, mit dem *User*-Button für den Benutzerbereich, dem *New*-Button zum Erstellen von neuen Entscheidungen, dem *Add*-Button zum Hinzufügen bereits existierender Entscheidungen und einem *Refresh*-Button zur Aktualisierung der Einträge. Jeder der Buttons verfügt dabei über ein eigenes, unverkennbares Icon, wodurch die Benutzerbedienung erleichtert werden soll.



Abbildung 4.2: Buttons des Hauptmenüs

In der rechten oberen Ecke der Hauptseite befinden sich die Einstellungen für die Internationalisierung (i18n), die einen applikationsweiten Sprachenwechsel ermöglicht. Da zum Zeitpunkt dieser Arbeit noch keine standardmäßige Unterstützung durch Sencha Touch vorhanden war, wurde die Umsetzung mittels selbst entwickelter Javascript-Library bewerkstelligt. Der Benutzer kann auf der Hauptseite durch Auswahl einer der Flaggen zwischen der deutschen und der englischen Sprache auswählen, die momentan aktive Sprache wird dabei mittels roter Umrandung der Flagge gekennzeichnet. Eine Erweiterung der Sprachen ist einfach zu bewerkstelligen, jedoch sollte dann der Modus des Sprachwechsels aus Übersichtlichkeit in ein eigenes Fenster ausgelagert werden.

4.2 Detailansicht

Die Detailansicht einer Gruppenentscheidungsaufgabe, wie in Abbildung 4.3 dargestellt, bietet neben dem Namen, der Beschreibung und dessen Teilnehmerliste einige Aktionsmöglichkeiten. Abhängig von der Phase in der sich eine Entscheidungsfindung befindet, bietet das *Toolbar*-Menü unterschiedliche Optionen an.



Abbildung 4.3: Detailansicht einer Gruppenentscheidungsaufgabe

Allen Teilnehmern steht bei einer noch nicht abgeschlossenen Entscheidung der *Fill Out*-Button zur Verfügung. Dieser ermöglicht es die Präferenzen abzugeben oder anzupassen. An seine Stelle tritt bei einer bereits abgeschlossenen Entscheidung der *Details*-Button, der bei Auswahl den Teilnehmern das finale Ergebnis präsentiert.

Der Administrator der Gruppenentscheidungsaufgabe bekommt auch noch einen *Edit*-Button zur Verfügung gestellt, der es ihm ermöglicht diese zu überarbeiten. Dieser Button ist nur solange vorhanden bis ein Teilnehmer seine Präferenzen abgegeben hat, danach ist eine

Änderung der Gruppenentscheidungsaufgabe nicht mehr möglich.

Haben die Teilnehmer ihre Präferenzen abgegeben, obliegt es dem Administrator die Gruppenentscheidungsaufgabe in die nächste Phase zu bringen. Dafür steht der *Advance*-Button zur Verfügung, der nach seiner Verwendung ebenfalls aus dem Menü verschwindet. Außerdem steht für den Administrator ein *Close*-Button bereit, um die Gruppenentscheidungsaufgabe zu finalisieren und das finale Ergebnis festzulegen.

Der Administrator hat auch die Möglichkeit eine Gruppenentscheidungsaufgabe aus dem System zu entfernen. Dazu dient der *Delete*-Button, der bei aktuellen sowie geschlossenen Aufgaben vorhanden ist.

Unter der Liste mit den Teilnehmern steht dem Administrator der Button *New Participant* zur Verfügung, mit dem neue Teilnehmer der Gruppenentscheidungsaufgabe hinzugefügt werden können. Nach der Auswahl des Buttons wird ein Formular angezeigt, das in Abbildung 4.4 dargestellt ist.

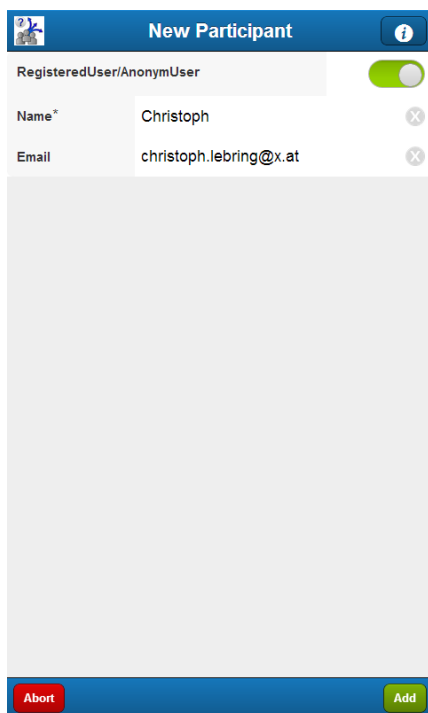


Abbildung 4.4: Teilnehmer hinzufügen

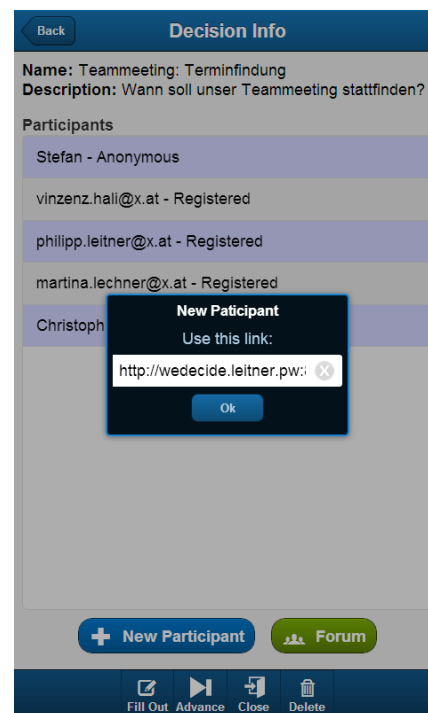


Abbildung 4.5: Link für anonymen Benutzer

In diesem Formular gibt es die Möglichkeit einen anonymen oder registrierten Benutzer der Gruppenentscheidungsfindung hinzuzufügen. Handelt es sich dabei um einen anonymen Benutzer, erscheint im Anschluss in einer Textbox ein Link zur Teilnahme, siehe Abbildung 4.5. Bei einem registrierten Benutzer wird dieser zur Teilnehmer-Liste hinzugefügt und per E-Mail verständigt.

Der *Forum*-Button ermöglicht es allen Teilnehmern miteinander in einem Forum zu kommunizieren, wie in Abbildung 4.6 dargestellt. Neben dem allgemeinen Forum einer Gruppenentscheidungsaufgabe, wird der Austausch zwischen den Teilnehmern mittels Kommentaren bei einzelnen Bereichen oder durch den Einsatz von Links und Dateien ermöglicht.

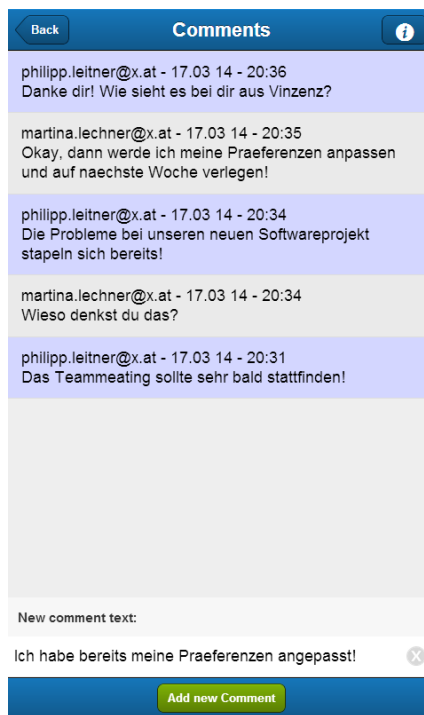


Abbildung 4.6: Forum einer Gruppenentscheidung

4.3 Benutzerbereich

Die Applikation ermöglicht, gleich wie die Web-Version von *WeDecide*, als anonymer oder registrierter Benutzer an einer Gruppenentscheidungsfindung teilzunehmen. Der Benutzerbereich dient dabei zur Regelung aller Funktionen eines registrierten Benutzers. Dazu zählen die Registrierung eines neuen Benutzers, das Anmelden, Abmelden und Ändern von Daten eines Benutzers sowie die Wiederherstellung des Passworts, falls dieses einmal vergessen werden sollte.

Um den Benutzerbereich anzeigen zu lassen, genügt die Auswahl des *User*-Buttons auf der Hauptseite. Wenn noch kein Benutzer in der Applikation angemeldet wurde erscheint das in Abbildung 4.7 dargestellte Anmeldeformular.

The screenshot shows the 'Login' screen. At the top, there is a blue header with a user icon and the title 'Login'. Below the header, there are two input fields: 'Username (email address)*' containing 'phillipp.leitner@x.at' and 'Password*' containing '*****'. A checkbox labeled 'stay logged in?' is checked. Below the password field, there is a link 'Or do you need to register?' and a green 'Register' button. At the bottom, there are three buttons: 'Cancel' (red), 'Forgot Password' (blue), and 'Login' (green).

Abbildung 4.7: Anmeldung eines Benutzers

The screenshot shows the 'Edit Profile' screen. At the top, there is a blue header with a user icon and the title 'Edit Profile'. Below the header, there are four input fields: 'Current Password*' containing 'phillipp.leitner@x.at', 'Password*' containing '*****', 'Retype Password*' containing '*****', and 'Phone Number' containing '0316/000000'. At the bottom, there is a note 'All fields marked with * must be filled out' and three buttons: 'Back' (red), 'Logout' (blue), and 'Ok' (green).

Abbildung 4.8: Profilverwaltung

Dort kann sich der Benutzer durch Eingabe einer gültigen Kombination aus Benutzername und Passwort am System anmelden. Dabei besteht die Möglichkeit mittels einer Checkbox

die Daten auf dem mobilen Gerät abzuspeichern und den Benutzer automatisch beim Start der Applikation anzumelden. Nachdem der Benutzer angemeldet wurde, erscheint anstelle des Anmeldeformulars im Benutzerbereich sein Profil, wie in Abbildung 4.8 sichtbar. Es besteht nun die Möglichkeit die Daten des Benutzers zu verändern und diese nach positiver Validierung am Server zu speichern. Weiters entfallen nach der Anmeldung in den vorhandenen Formularen die Felder für den anonymen Benutzer. Die bereits vorhandenen Gruppenentscheidungsaufgaben werden automatisch vom Server geladen und in einer Liste auf der Hauptseite angezeigt.

Falls der Benutzer noch keinen Account im System angelegt hat, kann dies durch Betätigen des *Register*-Buttons vorgenommen werden. Wie in Abbildung 4.9 zu sehen ist, gilt es dafür einige Daten anzugeben.

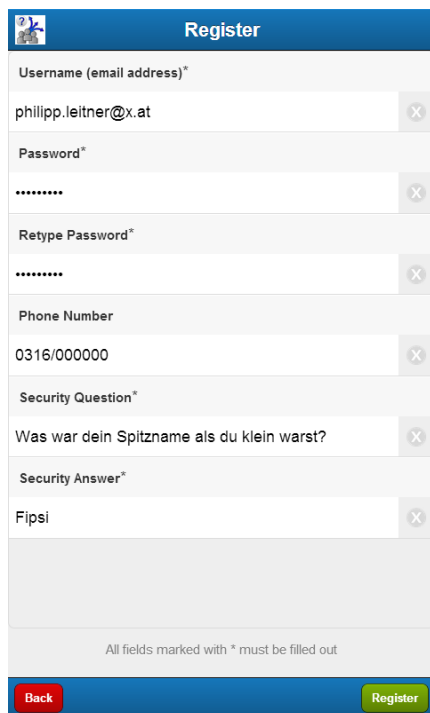


Abbildung 4.9: Registrierung eines Benutzers

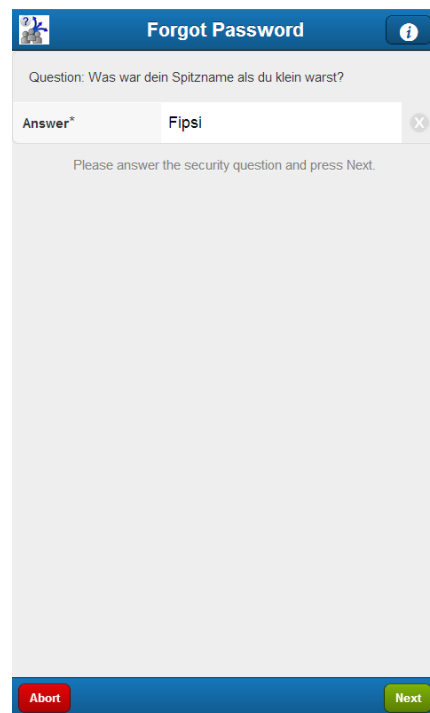


Abbildung 4.10: Zurücksetzen des Passworts

Der Benutzername ist in Form einer E-Mail-Adresse einzugeben, die später bei den Gruppenentscheidungsfindungen für die Kommunikation herangezogen wird. Die Eingabe des Passworts wird verdeckt getätigt und muss deshalb ein weiteres Mal bei der Passwortwieder-

derholung eingegeben werden. Das Feld für die Telefonnummer ist optional und muss daher nicht wie alle anderen Felder ausgefüllt werden. Die Sicherheitsfrage und deren Antwort dienen im Falle des Verlusts dem Zurücksetzen des Passworts, das Formular dafür wird in Abbildung 4.10 dargestellt. Durch Betätigen des *Register*-Buttons werden die eingegebenen Daten validiert und bei positivem Ergebnis der neue Benutzer angelegt.

Neben dem registrierten Benutzer besteht die Möglichkeit eine Gruppenentscheidungsaufgabe über einen anonymen Benutzer zu erstellen. Dieser Benutzer wird dabei automatisch auf dem mobilen Gerät in einem lokalen Speicher abgelegt und bei jeder Aktualisierung vom Server mitgeladen. Durch Auswahl des *Add*-Buttons auf der Hauptseite öffnet sich ein Formular um einen anonymen Benutzer über seine Universally Unique Identifier (UUID) hinzuzufügen. Wird eine Gruppenentscheidungsaufgabe, die von einem anonymen Benutzer erstellt wurde oder an der dieser teilgenommen hatte gelöscht, so wird der anonyme Benutzer vom lokalen Speicher entfernt.

An dieser Stelle sollte angemerkt werden, dass die Verwendung eines anonymen Benutzers auf mobilen Geräten wenig Sinn ergibt, da die Benutzer über die Verwendung ihres Gerätes identifiziert werden könnten. Jedoch soll die mobile Applikation den vollen Umfang der Web-Version von *WeDecide* widerspiegeln und unterstützt daher diese Möglichkeit.

4.4 Ablauf der Gruppenentscheidungsaufgaben

Der Ablauf der Gruppenentscheidungsaufgaben kann in drei Abschnitte aufgeteilt werden. Der erste Schritt ist die Modellierung der Gruppenentscheidungsaufgabe, wie im Diagramm in Abbildung 4.11 sichtbar. Dabei wird anhand der fünf verschiedenen Farben der Ablauf der unterschiedlichen Arten von Gruppenentscheidungsaufgaben skizziert: „Antworten finden“, „Wähle eine Alternative“, „Suche & wähle eine Alternative“, „Einen Termin finden“ und „Anforderungen priorisieren“.

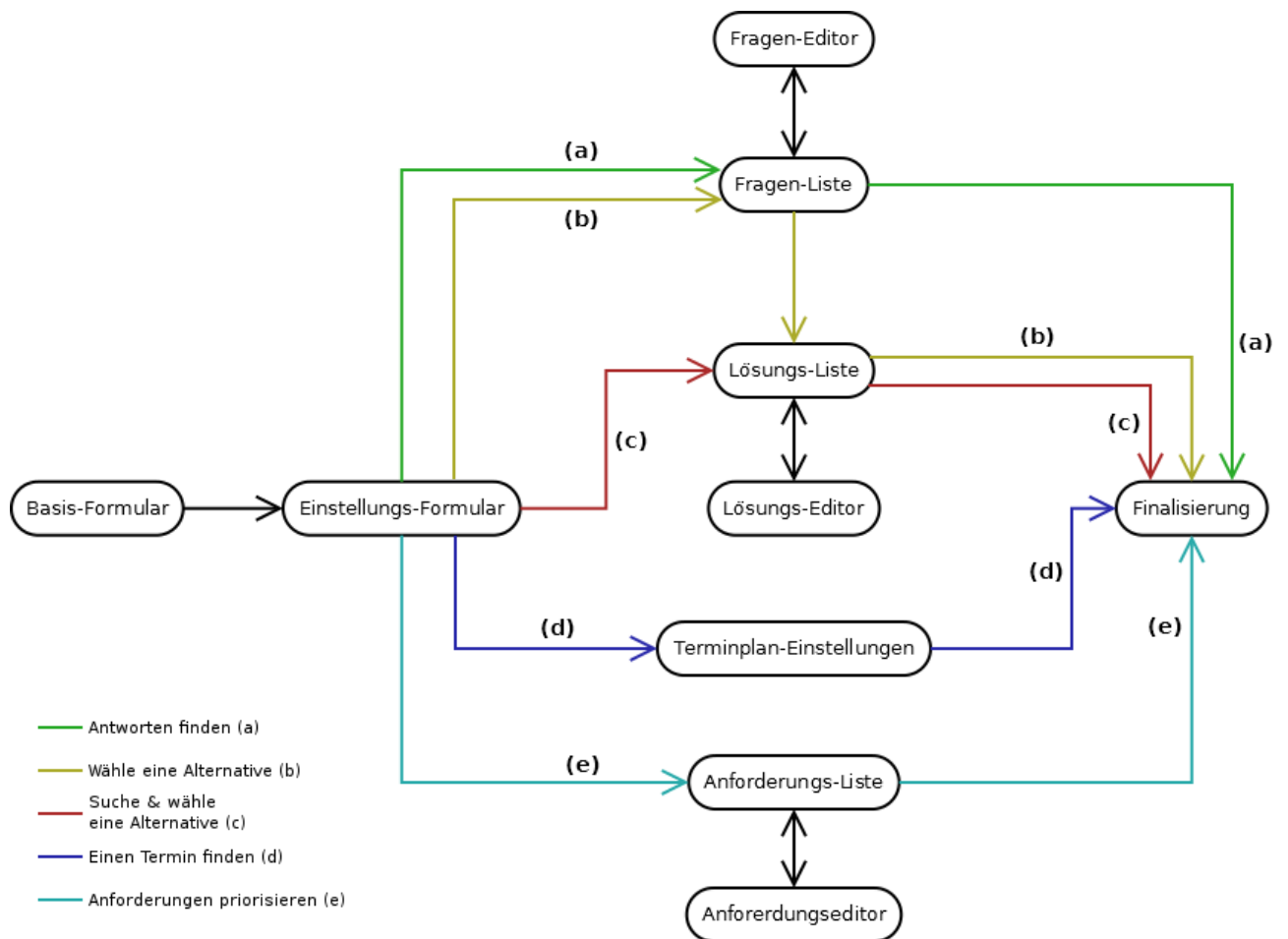


Abbildung 4.11: Diagramm zum Ablauf der Modellierung einer Gruppenentscheidungsaufgabe

Die Modellierung zählt zu den Grundpfeilern einer Gruppenentscheidungsaufgabe und bestimmt durch die Wahl der Einstellungen den kompletten Ablauf der Entscheidungsfindung. Als nächstes werden Benutzer zur Gruppenentscheidungsfindung eingeladen und können ihre Präferenzen abgeben, dabei wird zwischen zwei Phasen unterschieden. In *Phase 1* einer Gruppenentscheidungsaufgabe soll der Benutzer seine Präferenzen abgeben, ohne die Präferenzen der anderen Benutzer zu kennen oder eine Gruppenentscheidungshilfe zur Verfügung gestellt zu bekommen. Durch dieses Vorgehen wird gewährleistet, dass die Benutzer nicht von Beginn an von anderen Benutzern in ihren Entscheidungen beeinflusst werden. Wurden die Präferenzen der Benutzer aufgenommen oder ist eine vom Administrator bestimmte Zeitspanne verstrichen, wird *Phase 2* der Entscheidungsaufgabe gestartet. Erst jetzt werden den Benutzern die Ergebnisse des Gruppenentscheidungsalgorithmus mitgeteilt. Abhängig von den während der Modellierung gewählten Einstellungen, werden nun auch die Präferenzen der anderen Benutzer angezeigt. In *Phase 2* können die Benutzer ihre Präferenzen überarbeiten. Im letzten Schritt der Finalisierung obliegt es dem Administrator die Gruppenentscheidungsaufgabe abzuschließen und den Benutzern zu präsentieren. Die einzelnen Schritte bei den unterschiedlichen Arten von Gruppenentscheidungsaufgaben werden in Abbildung 4.12 in Phasen und deren Übergängen dargestellt.

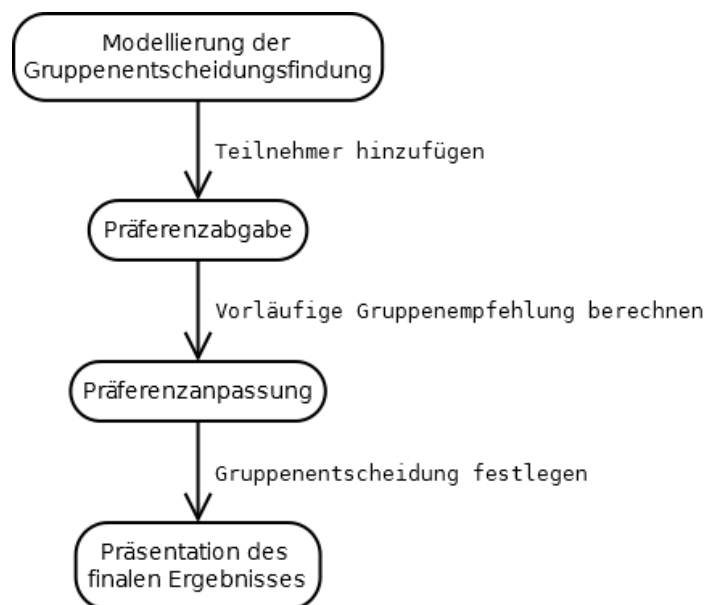


Abbildung 4.12: Phasendiagramm des Gruppenentscheidungsprozesses

4.5 Antworten finden

Das Szenario “Antworten finden“ sollte dann zum Einsatz kommen, wenn es um das Finden von gemeinsamen Antworten auf Fragen geht. Dabei stehen die Attribute im Vordergrund und auf eine Spezifizierung der Lösungsmenge wird bewusst verzichtet. In diesem Kapitel wird anhand des Beispiels der Konfiguration eines Computers, der Ablauf innerhalb der Applikation beschrieben. Dabei wird es in die drei Unterkapitel Modellierung, Teilnahme und Finalisierung aufgeteilt.

4.5.1 Modellierung

Am Beginn der Modellierung steht das Basis-Formular (Abbildung 4.13), welches für alle Arten von Gruppenentscheidungsaufgaben gleich ist.

The image shows a mobile application interface for creating a new decision. The title bar is blue and contains the text "New Decision" and an information icon. Below the title bar, there are three main input sections. The first is "Name of group decision*" with a text input field containing "Konfigurator für den Familiencomputer" and a clear button (X). The second is "Type*" with a dropdown menu currently showing "Find answers". The third is "Description" with a text area containing "Die Familie braucht einen neuen Computer, was dieser können soll, wird hier entschieden werden." and a clear button (X). At the bottom of the form, there is a red "Abort" button and a green "Next" button. A small note at the bottom of the form states "All fields marked with * must be filled out".

Abbildung 4.13: Basis-Formular einer Modellierung

Dabei wird ein Name und optional eine Beschreibung der Gruppenentscheidungsaufgabe festgelegt. Über ein Auswahlfeld wird hier auch die Gruppenentscheidungsart festgelegt. Nach dessen Auswahl öffnet sich ein *Picker* mit den Einträgen zu den Arten der möglichen Gruppenentscheidungsaufgaben, wie in Abbildung 4.14 dargestellt. Jeder Eintrag enthält zusätzlich zum Namen auch ein für ihn spezifisches Icon, das sich dann auf der Hauptseite bei der Ansicht der vorhandenen Gruppenentscheidungsaufgaben wiederfindet und so zur Übersichtlichkeit beiträgt.

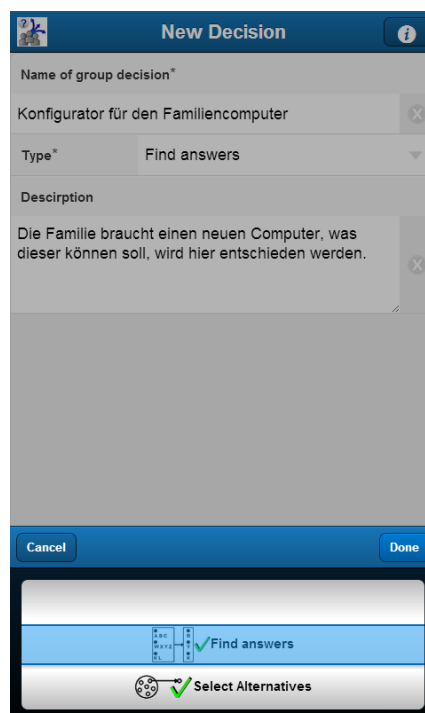
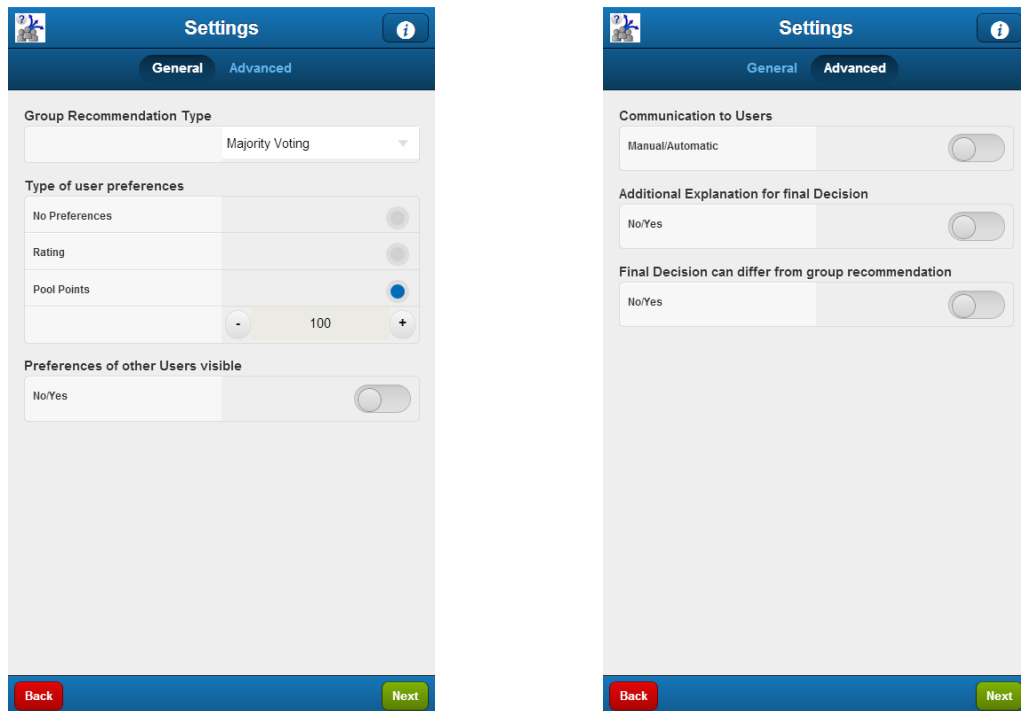


Abbildung 4.14: Picker für die Gruppenentscheidungsart

Nachdem bei Art der Gruppenentscheidungsaufgabe “Antworten finden“ ausgewählt wurde, geht es weiter mit den spezifischen Einstellungen der Aufgabe. Diese wurde der Übersicht halber in einen allgemeinen und einen erweiterten Teil unterteilt, wie Abbildung 4.15 zeigt.

Im allgemeinen Bereich hat der Ersteller die Möglichkeit die Art des verwendeten Gruppenempfehlungsalgorithmus zu wählen. Wird das Auswahlfeld betätigt, wird abermals ein *Picker* angezeigt, in dem der Ersteller benutzerfreundlich aus den vorhandenen Algorithmen auswählen kann.



(a) Allgemeine Einstellungen

(b) Erweiterte Einstellungen

Abbildung 4.15: Spezifische Einstellungen des „Antworten finden“ Szenarios

Die zweite Einstellungsmöglichkeit, bei der der Ersteller bestimmen kann wie die gewählten Antworten bewertet werden können, befasst sich mit der Art der Benutzerpräferenzen. Im Fall des Szenarios der Konfigurierung wurde hier *Pool Points* ausgewählt, dabei erscheint zusätzlich ein *Spinnerfield*, mit dem die Anzahl der dabei zu verwendeten Punkte angegeben werden müssen. Diese Punkte werden von den Teilnehmern während der Präferenzabgabe auf die Fragen verteilt, wobei gilt, je höher die vergebenen Punkte sind, desto wichtiger ist die Frage für den Teilnehmer.

Die nächste Einstellungsoption befasst sich mit der Sichtbarkeit der Präferenzen anderer Benutzer. Bei der Standardeinstellung, die in diesem Beispiel beibehalten wurde, wird auf den Austausch der Präferenzen unter den Benutzern verzichtet.

Die erweiterten Einstellungen der Gruppenentscheidungsaufgabe bieten zusätzliche Optionen, die aber grundsätzlich bei den Grundeinstellungen belassen werden können. Als erstes kann die Art und Weise der Kommunikation der Benutzer eingestellt werden, dabei wird zwischen *Manuell* und *Automatisch* unterschieden. Die Grundeinstellung ist hier *Manuell*,

wobei der Austausch von Informationen der Gruppenentscheidungsaufgabe dem Ersteller und Administrator obliegt. Bei der Wahl der automatischen Kommunikation werden die Teilnehmer über Statusänderungen bei der Gruppenentscheidungsaufgabe über das System mittels E-Mail informiert.

Als nächstes kann der Ersteller bestimmen, ob er am Schluss der Entscheidungsfindung eine zusätzliche Erklärung in Form eines kurzen Textes abgeben möchte. Diese Option sollte vor allem dann gewählt werden, wenn der Administrator die finale Entscheidung unabhängig von der Empfehlung des Systems abgeben möchte. Dies ermöglicht die nächste Option der erweiterten Einstellungen, auf die in diesem Beispiel jedoch verzichtet wird, da sich der Administrator der Gruppenentscheidungsaufgabe und der Empfehlung des Systems anschließen wird.

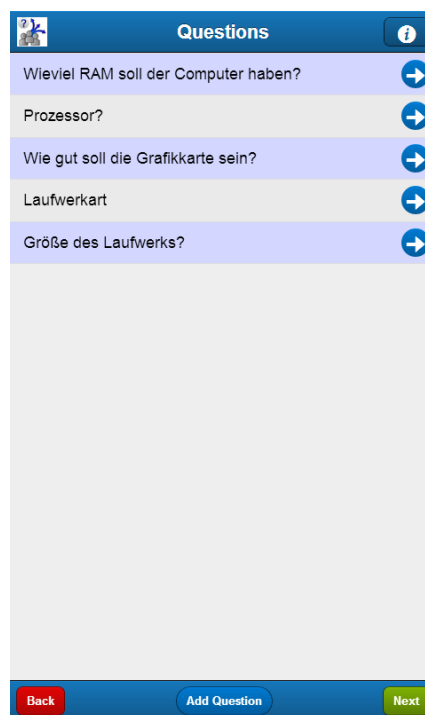


Abbildung 4.16: Fragenliste des „Antworten finden“ Szenarios

Nachdem die Einstellungen abgeschlossen sind, kann der Ersteller mit der Zusammenstellung der Fragen beginnen, auf welche die späteren Teilnehmer ihre Präferenzen abgeben werden. Dafür steht eine Fragenliste, wie Abbildung 4.16 dargestellt, zur Verwaltung der gesammelten

Fragen zur Verfügung. Neue Fragen können über den *Add Question*-Button hinzugefügt sowie bereits vorhandene Fragen durch Auswahl in der Liste editiert werden.

Der sich dabei öffnende Editor (Abbildung 4.17) besteht wiederum aus zwei Bereichen. Im allgemeinen Bereich muss der Ersteller einen Text für die Frage eintragen und dazu passende Antworten definieren. Der erweiterte Bereich bietet Einstellungsmöglichkeiten zu der Art der Frage, bei der festgelegt wird ob die Benutzer nur eine Antwort oder aber auch mehrere auswählen können. Als zweite Einstellungsoption steht die Art der Antwort, die entweder numerischer oder textueller Natur ist. Als letztes hat der Ersteller die Möglichkeit eine Frage als optional einzustufen, wodurch die Teilnehmer diese Frage bei der Präferenzabgabe übergehen können, wenn sie dies wünschen.

(a) Allgemeiner Bereich

(b) Erweiterter Bereich

Abbildung 4.17: Frageneditor des „Antworten finden“ Szenarios

Mittels des Buttons *Save Question* wird die Frage nach erfolgreicher Validierung gespeichert und die komplette Fragenliste ist wieder sichtbar. Nachdem mindestens eine Frage in der Liste

vorhanden ist, kann durch Auswahl des *Next*-Buttons mit der Finalisierung der Modellierung fortgesetzt werden.

Die Finalisierung der Modellierung, dargestellt in Abbildung 4.18, ist ebenso wie das Basis-Formular bei allen Arten der Gruppenentscheidungsaufgaben gleich. Es wird die Möglichkeit geboten die soeben erstellte Gruppenentscheidungsaufgabe für andere als Template zur Verfügung zu stellen. Dafür kann der Ersteller *Tags* zu einer Liste hinzufügen, die mit der Gruppenentscheidungsaufgabe zu tun haben, da es in diesem Fall um eine Computerkonfiguration geht, wurden Tags wie „Computer“, „Konfigurator“ oder aber auch „Grafikkarte“ verwendet. Als Privatsphäreneinstellung kann ausgewählt werden ob der Ersteller mit seinem Namen angezeigt werden möchte.

The screenshot shows a 'Finalization' form with the following elements:

- Header: Blue bar with a user profile icon and the text 'Finalization'.
- Toggle 1: 'Should it be possible that the Group Decision Problem is reused by others?' (checked).
- Toggle 2: 'Should your Name, as the creator of the Decision, be visible?' (checked).
- Section: 'Tags' with a list of items: 'Computer', 'Konfigurator', 'Arbeitspeicher', 'Prozessor', 'Festplatte', 'Grafikkarte'.
- Input: 'Name' field with a green '+ Add' button.
- Footer: Red 'Back' button and green 'Accept' button.

Abbildung 4.18: Finalisierung einer Modellierung

4.5.2 Teilnahme

Nach erfolgreicher Erstellung der Gruppenentscheidungsaufgabe ist der Administrator in der Lage anonyme sowie registrierte Benutzer zu der Entscheidungsfindung einzuladen. Dies wurde bereits in Kapitel 4.1, das sich mit der Verwaltung bereits existierender Gruppenentscheidungsaufgabe befasst, erläutert.

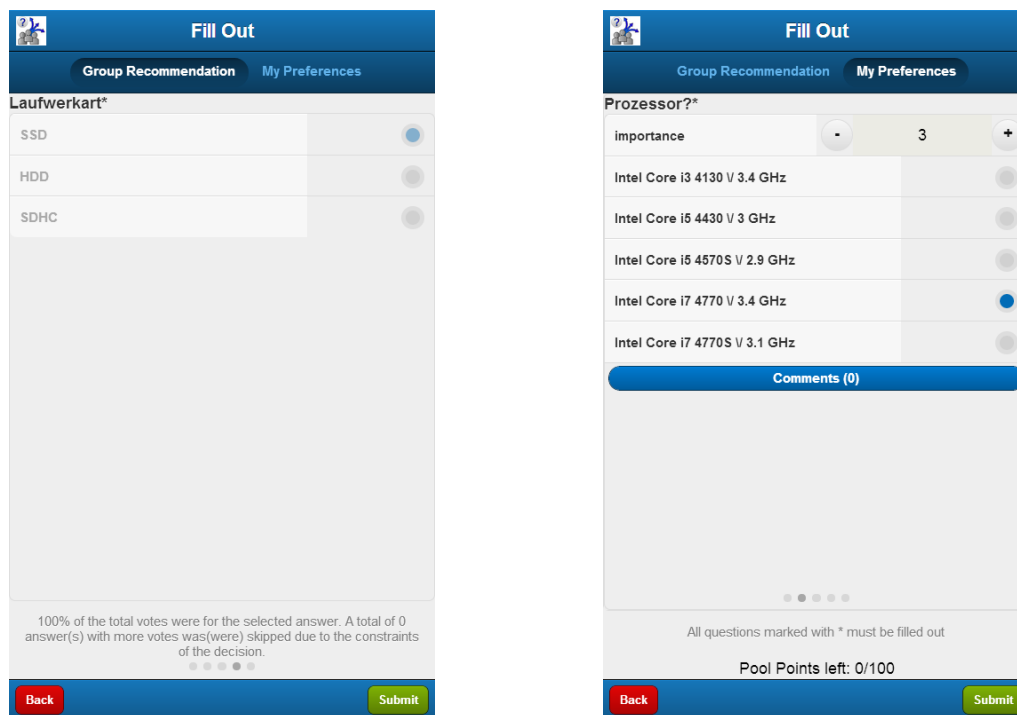
The screenshot shows a mobile application interface for a 'Fill Out' form. The question is 'Wieviel RAM soll der Computer haben?*' with an importance of 7. The options are 2 GB, 4 GB, 8 GB, 12 GB, and 16 GB. The 4 GB option is selected. Below the options is a 'Comments (0)' section. At the bottom, there is a 'Pool Points left: 53/100' indicator and 'Back' and 'Submit' buttons.

Abbildung 4.19: Phase 1 des „Antworten finden“ Szenarios

Nachdem die Teilnehmer zur Gruppenentscheidungsfindung hinzugefügt wurden, können diese ihre Präferenzen abgeben. Die während der Modellierung erstellten Fragen werden in der ersten Phase der Teilnahme als Formular (siehe Abbildung 4.19) angezeigt und bieten dem Teilnehmer die Möglichkeit seine Präferenzen abzugeben, ohne dabei durch das Wissen der Präferenzen der anderen Teilnehmer beeinflusst zu werden. Die Fragen werden dabei zur besseren Übersicht mittels einer *Caroussel-View* angezeigt, in der mittels Wischgeste zwischen den Fragen gewechselt werden kann. Die Art der Eingabe der Präferenzen ist dabei abhängig von der bei der Modellierung gesetzten Einstellungen *Type of user preferences*.

Da während der Modellierung *Pool Points* gewählt wurde, erscheint bei jeder Frage ein zusätzliches *Spinnerfield*, in dem die Teilnehmer durch Punkteabgabe die Relevanz der Frage bewerten. Am unteren Ende werden die Punkte, die noch auf die Fragen verteilt werden können, angezeigt.

Nachdem die Teilnehmer ihre Präferenzen abgegeben haben oder eine ausreichende Zeitspanne vergangen ist, obliegt es dem Administrator die Gruppenentscheidungsfindung in die nächste Phase fortschreiten zu lassen. Dies geschieht, wie bereits im Kapitel 4.1 über die Verwaltung der Gruppenentscheidungsaufgabe, in dessen Detailansicht. In Phase 2 der Gruppenentscheidungsfindung wird mittels einer *Tabbar* ein zweites Fenster neben dem Formular zur Präferenzabgabe, welches unter (b) von Abbildung 4.20 zu sehen ist, eingeführt.



(a) Gruppenempfehlung

(b) Präferenzabgabe

Abbildung 4.20: Phase 2 des „Antworten finden“ Szenarios

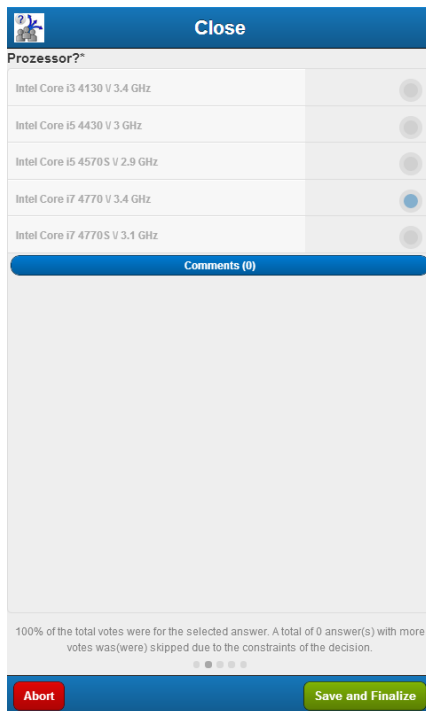
Dieses Fenster besteht ebenfalls aus einem Formular mit den Fragen in einer *Carousel-View*, jedoch sind diese bereits mit der vom Algorithmus berechneten Gruppenempfehlung

ausgefüllt, deaktiviert und dienen nur zur Informationsvermittlung für die Teilnehmer, wie unter (a) in Abbildung 4.20 dargestellt wird. Bei jeder der Fragen steht am unteren Ende eine Beschreibung über das Zustandekommen der Empfehlung. Da bei der Modellierung die Sichtbarkeit der anderen Benutzerpräferenzen deaktiviert wurde, werden diese auch nicht angezeigt. Dem Teilnehmer steht es frei seine eigenen Präferenzen mit der Gruppenempfehlung zu vergleichen und diese falls notwendig zu ändern.

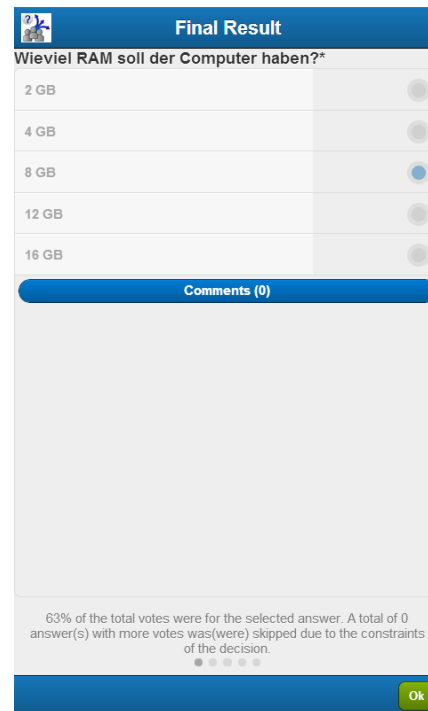
Während der gesamten Zeit steht den Teilnehmern ein Forum zur Verfügung, wie es in Kapitel 4.1 bereits beschrieben wurde. Weiters besteht die Möglichkeit Kommentare bei den einzelnen Fragen zu hinterlassen, die dann von den anderen Teilnehmern gelesen werden können. Dafür bietet sich der *Comments*-Button bei der Präferenzabgabe an, bei dem zur Steigerung der Benutzerfreundlichkeit die Anzahl der bereits vorhandenen Kommentare angezeigt wird.

4.5.3 Finalisierung

Nachdem die Teilnehmer auch in der zweiten Phase genug Zeit für Anpassungen ihrer Präferenzen und den Dialog miteinander hatten, kann der Administrator die Finalisierung der Gruppenentscheidungsfindung durchführen. Die Gruppenempfehlung wird dabei neu berechnet und dem Administrator, wie in der linken Ansicht *Finalisierung* in Abbildung 4.21 abgebildet, präsentiert. Da während der Modellierung festgelegt wurde, dass sich in unserem Beispiel das finale Ergebnis nicht von der Empfehlung unterscheiden kann, bleibt dem Administrator nur die Möglichkeit das Ergebnis hinzunehmen und die Entscheidungsfindung durch das Auswählen des *Save and Finalize*-Buttons abzuschließen.



(a) Finalisierung



(b) Finales Ergebnis

Abbildung 4.21: Abschluss des „Antworten finden“ Szenarios

Nach der Finalisierung wird die Gruppenentscheidungsaufgabe aus der Liste mit den aktuellen Gruppenentscheidungen in die Liste der geschlossenen Entscheidungen verschoben. Die Teilnehmer können nun ihre Präferenzen nicht mehr ändern, jedoch über den *Details*-Button der Detailansicht sich das finale Ergebnis präsentieren lassen, zu betrachten unter *Finales Ergebnis* in Abbildung 4.21. Abermals wird bei den Fragen eine Beschreibung über das Zustandekommen der Bewertung angezeigt.

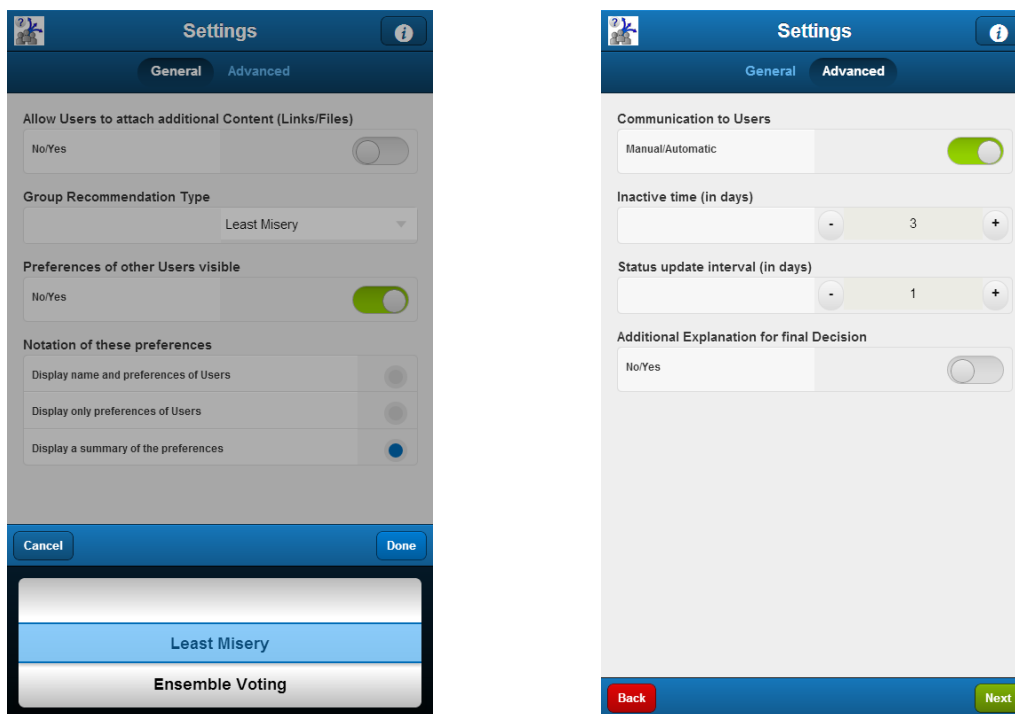
4.6 Wähle eine Alternative

Das “Wähle eine Alternative“ Szenario gehört ebenfalls wie das vorherige zu den *Choice Decision* Szenarien. In diesem Fall werden den Teilnehmern mehrere Alternativen zur Auswahl vorgelegt und dabei auf Attribute zur Gruppenentscheidungsunterstützung verzichtet. Da es daher keine Vorselektierung vor der Präferenzabgabe gibt, eignet sich dieses Szenario entweder für eine Aufgabenstellung mit wenigen Alternativen oder aber wenn eine Vorselektierung schwer oder überhaupt nicht möglich wäre. In diesem Abschnitt wird als konkretes Beispiel die Suche nach einem passenden Brettspiel für einen Spieleabend verwendet.

4.6.1 Modellierung

Die Modellierung dieses Szenarios läuft ähnlich der Modellierung eines “Antworten finden“ Szenarios ab, die in Kapitel 4.5.1 nachzulesen ist. Zuerst gilt es wieder das Basis-Formular auszufüllen, nur wird dieses Mal “Wähle eine Alternative“ bei der Art der Gruppenentscheidungsaufgabe gewählt. Danach geht es weiter mit den spezifischen Einstellungen, die wiederum in einen allgemeinen und einen erweiterten Teil aufgeteilt wurden. Die Oberfläche beider Teile ist in Abbildung 4.22 dargestellt.

Die Einstellungen ähneln denen der vorherigen Gruppenentscheidungsaufgabe des Typs „Antworten finden“. Zu den Neuerungen gehört die erste Einstellungsoption im allgemeinen Teil, dabei erlaubt der Ersteller den Benutzern das Hinzufügen von Links oder Dateien und bietet somit dem Teilnehmernaustausch weitere Möglichkeiten. Die zweite Einstellungsoption dient der Art des verwendeten Gruppenempfehlungsalgorithmus, die bereits aus dem vorherigen Szenario bekannt ist. Unter *Allgemeine Einstellungen* in Abbildung 4.22 wird die Auswahl mittels *Picker* gerade vollzogen. Die letzte Einstellungsmöglichkeit befasst sich wie in der vorherigen Gruppenentscheidungsaufgabe mit der Sichtbarkeit der Präferenzen der anderen Benutzer. In diesem Beispiel wurde sie jedoch aktiviert, wodurch eine weitere Einstellungsoption zur Darstellung der Präferenzen angezeigt wird. Die dabei ausgewählte Option wird während der Teilnahme dazu führen, dass eine Zusammenfassung der Präferenzen der Teilnehmer bei der Gruppenempfehlung angezeigt wird.



(a) Allgemeine Einstellungen

(b) Erweiterte Einstellungen

Abbildung 4.22: Spezifische Einstellungen des „Wähle eine Alternative“ Szenarios

Im erweiterten Teil gibt es die ebenfalls bereits aus dem letzten Kapitel bekannte Option um die Art und Weise der Kommunikation der Benutzer einzustellen. Im angegebenen Beispiel wurde in diesem Beispiel jedoch *Automatisch* ausgewählt, wodurch zwei *Spinnerfields* sichtbar werden. Das erste Feld ermöglicht die Auswahl der inaktiven Zeit in Tagen und das zweite Feld gibt in Tagen das Intervall an, in dem Status-Updates versendet werden. Als letzte Einstellung besteht die Möglichkeit am Schluss der Entscheidungsfindung eine zusätzliche Erklärung in Form eines kurzen Textes anzugeben, was jedoch im vorliegenden Beispiel deaktiviert bleibt.

Nachdem die Einstellungen abgeschlossen sind, kann der Ersteller mit dem Füllen der Alternativenliste, die unter (a) in Abbildung 4.23 zu sehen ist, beginnen.

Neue Alternativen können durch die Auswahl des *Add new Solution*-Buttons hinzugefügt werden. Dabei öffnet sich eine *Messagebox* mit einem Textfeld für den Namen der Alternative, dargestellt in (b) von Abbildung 4.23. Nach der Eingabe und dem betätigen des *Ok*-Buttons, erscheint die neue Alternative in der Liste. Bereits vorhandene Alternativen können durch

Auswahl in der Liste wieder entfernt werden.

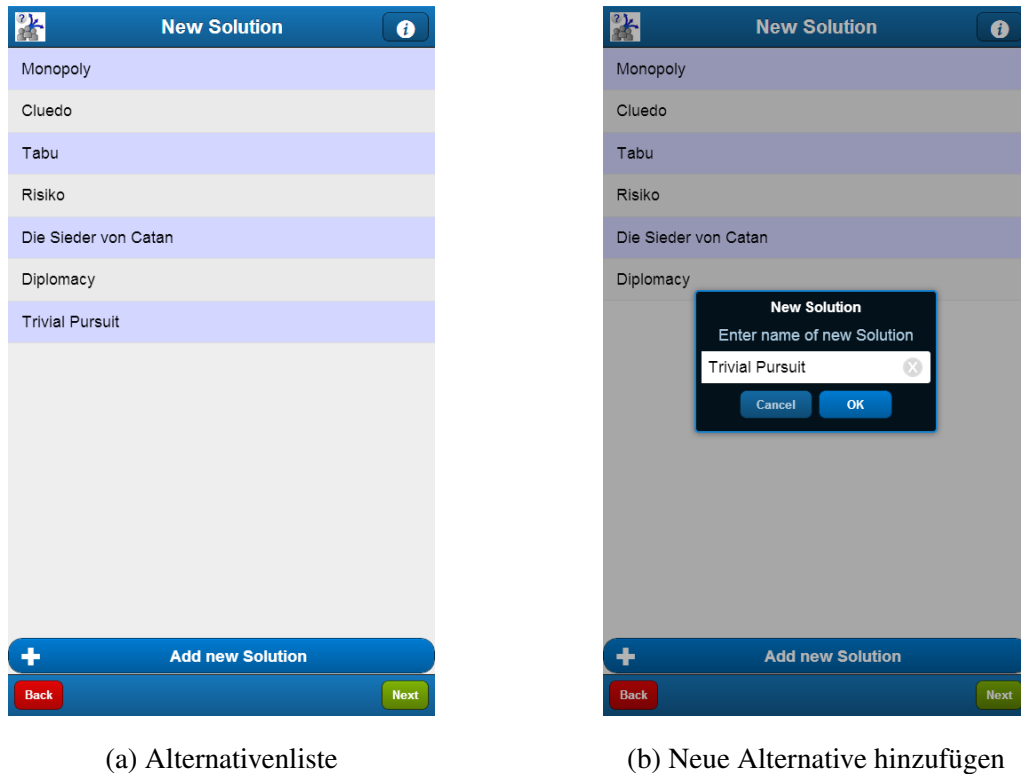


Abbildung 4.23: Modellierung der Alternativen des „Wähle eine Alternative“ Szenarios

Weitere Einstellungen gibt es in diesem Szenario nicht, daher kann nach Eingabe von mindestens zwei Alternativen mit der Finalisierung fortgefahren werden. Da die Finalisierung, deren Oberfläche im vorherigen Kapitel 4.5.1 in Abbildung 4.18 zu sehen ist, sich in den einzelnen Arten der Gruppenentscheidungsaufgaben nicht unterscheidet, wird diese hier nicht erneut erörtert.

4.6.2 Teilnahme

Nach der erfolgreichen Erstellung der Gruppenentscheidungsaufgabe obliegt es dem Administrator die Teilnehmer der Gruppenentscheidungsfindung hinzuzufügen. Danach kann jeder der Teilnehmer seine Präferenzen abgeben. Dabei gibt es wieder die erste Phase, in der

die Teilnehmer ohne Einwirkung ihre Präferenzen abgeben können und Phase 2, bei der die Teilnehmer die vorläufige Gruppenempfehlung erhalten und ihre Präferenzen gegebenenfalls anpassen können.

In Phase 1 werden die Teilnehmer aufgefordert ihre Präferenzen in Form einer Wertung von 1 bis 5 Sternen bei den vorhandenen Alternativen abzugeben. Abbildung 4.24 zeigt die Liste der vorher definierten Brettspiele und die abgegebenen Sterne des Teilnehmers. Da auf das Hinzufügen von Links oder Dateien verzichtet wurde, haben die Benutzer nur die Möglichkeit einer Interaktion miteinander über die Kommentarfunktion. Die Präferenzen der anderen Benutzer sind in dieser Phase der Gruppenentscheidungsfindung noch nicht sichtbar. Die Präferenzabgabe wird durch das Betätigen des *Submit*-Buttons abgeschlossen.

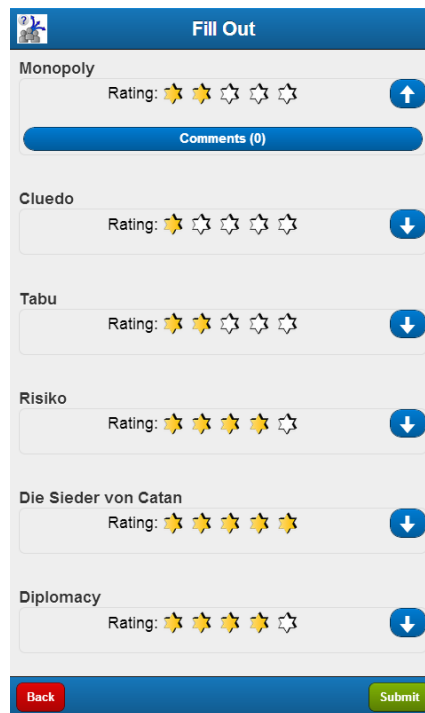
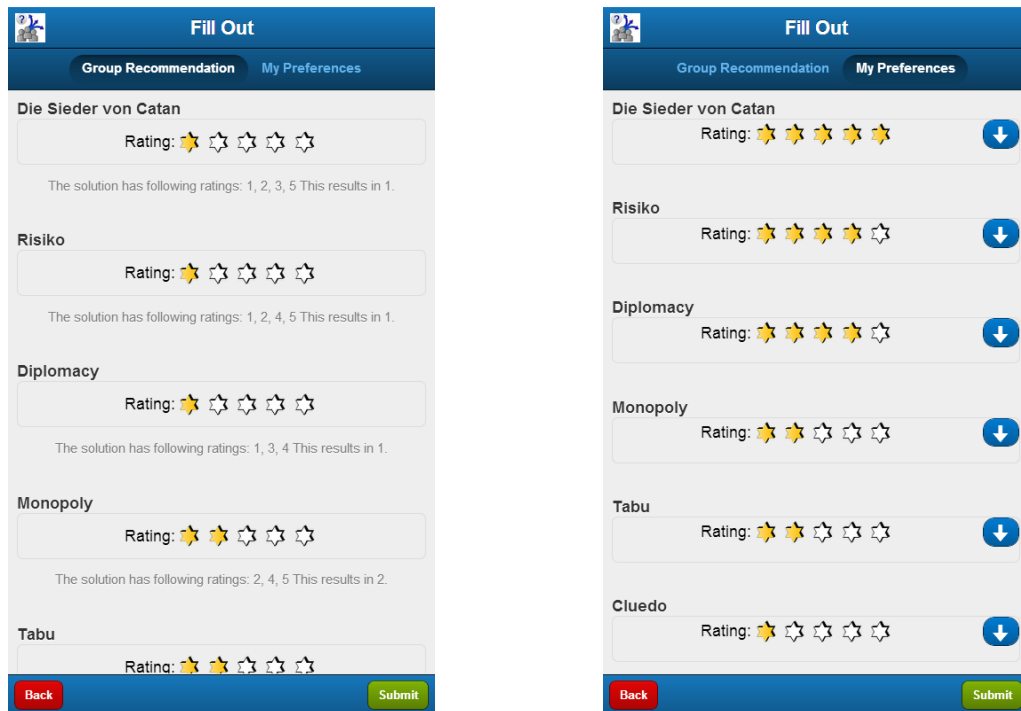


Abbildung 4.24: Phase 1 des „Wähle eine Alternative“ Szenarios

Dem Administrator obliegt abermals die Aufgabe die Gruppenentscheidungsaufgabe nach einer angemessenen Zeit in die nächste Phase fortschreiten zu lassen. In Phase 2 erscheint wieder in der *Tabbar* ein eigener *Tab* für die vom Algorithmus berechnete, vorläufige Gruppenempfehlung. Die Oberfläche der Gruppenempfehlung ist in Abbildung 4.25 abgebildet

und präsentiert die Alternativenliste mit der berechneten Wertung in Form der bekannten 5-Sterne-Skala. Bei jedem Eintrag steht die Zusammenfassung der Präferenzabgaben, der Teilnehmer, wie es während der spezifischen Einstellung in der Modellierung gewollt war. Abermals steht den Teilnehmern frei, im zweiten *Tab* ihre in Phase 1 abgegebenen Präferenzen anzupassen.



(a) Gruppenempfehlung

(b) Präferenzanpassung

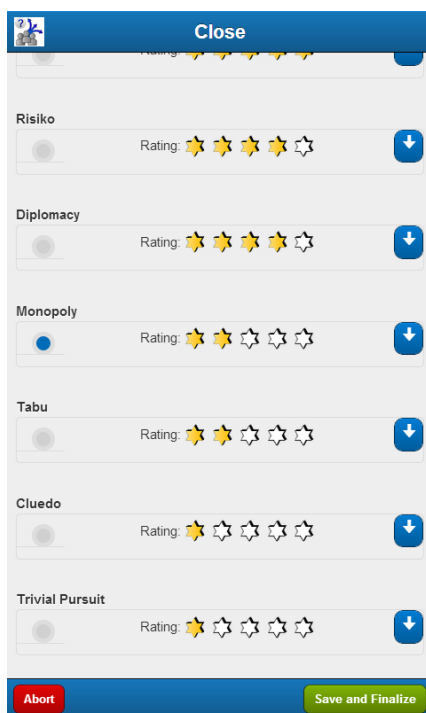
Abbildung 4.25: Phase 2 des „Wähle eine Alternative“ Szenarios

Während der Teilnahme steht neben der Kommentarfunktion auch wieder das allgemeine Forum der Guppenentscheidungsaufgabe zum Austausch zwischen den Teilnehmern bereit. Dies wurde bereits im Kapitel 4.1 erklärt und dargestellt.

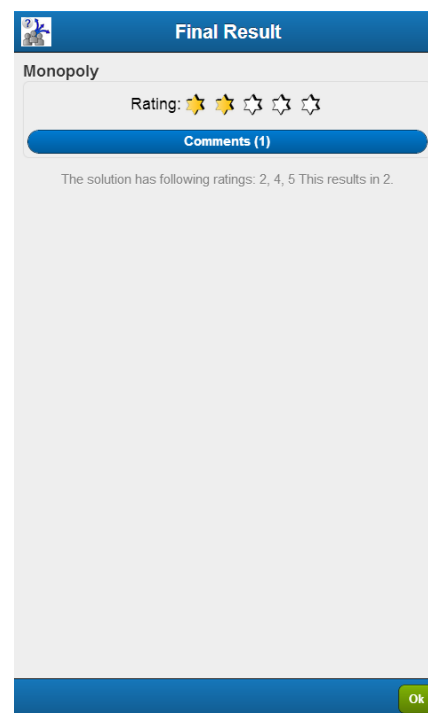
4.6.3 Finalisierung

Am Abschluss der Gruppenentscheidungsfindung steht wieder die Finalisierung. Dabei wird vom Administrator erwartet, dass er nach einer angemessenen Zeit, in der die Teilnehmer ihre

Präferenzen in Phase 2 überarbeiten können, das finale Ergebnis der Gruppenentscheidungsaufgabe festlegt. Unter (a) in Abbildung 4.26 ist die Ansicht des Finalisierungsformulars abgebildet. Es werden alle Alternativen mit ihrer abschließend berechneten Gruppenempfehlungsbewertung angezeigt. Dem Administrator obliegt jetzt die Aufgabe eine der Alternativen, mit möglichst guten oder sogar besten Bewertungen auszusuchen. Unterstützt wird er dabei von den Erklärungen über das Zustandekommen der Bewertung, dass er mittels Auswahl des Dropdown-Menü bei den einzelnen Alternativen öffnen kann. Zur Auswahl des finalen Ergebnisses wählt er eine der *Checkboxes* auf der linken Seite der Alternativen aus. Da während der Modellierung auf die zusätzliche Erklärung in Form eines kurzen Textes verzichtet wurde, wird diese Option auch nicht angezeigt. Nach Abschluss durch Betätigung des *Speichern und Finalisieren*-Buttons werden die Teilnehmer über die Beendigung informiert und die Gruppenentscheidungsaufgabe wandert im Hauptmenü unter *Geschlossene Entscheidungen*. Dort kann dann das finale Ergebnis von allen Teilnehmern begutachtet werden, wie unter (b) in Abbildung 4.26 dargestellt wird.



(a) Finalisierung



(b) Finales Ergebnis

Abbildung 4.26: Abschluss des „Wähle eine Alternative“ Szenarios

4.7 Suche & wähle eine Alternative

Das “Suche & wähle eine Alternative“ Szenario kommt zum Einsatz, wenn Attribute dazu verwendet werden den Lösungsraum einzuschränken. Dafür werden vom Ersteller der Gruppenentscheidungsaufgabe Fragen zusammengestellt, durch die die Teilnehmer ihre Präferenzen ermitteln können. In weiterer Folge wird dann die Liste mit den Alternativen den Teilnehmern präsentiert, wobei Alternativen die ihren Vorstellungen eher entsprechen in der Liste weiter oben gereiht werden.

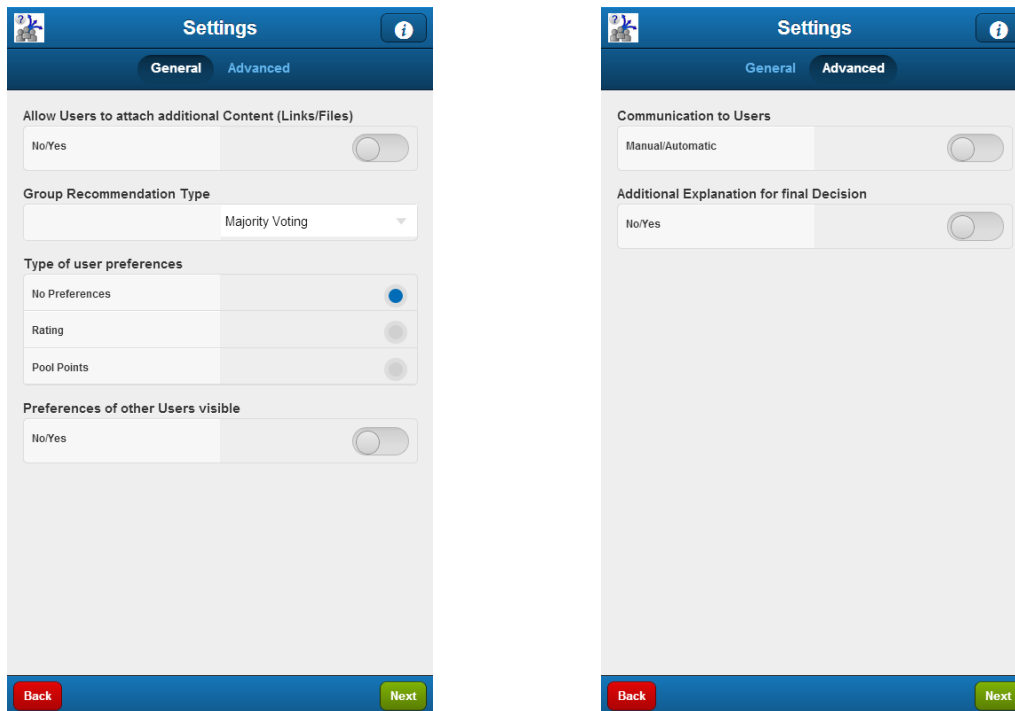
Dieses Szenario bietet dadurch eine Kombination der beiden zuvor erwähnten Szenarien und wird in diesem Abschnitt anhand des Beispiels der Konfiguration eines Autos, wie sie im Internet öfters zu finden ist, erörtert.

4.7.1 Modellierung

Wie bei allen Szenarien ist der Ausgangspunkt der Modellierung wieder das Basis-Formular, dass in Kapitel 4.5.1 in Abbildung 4.13 dargestellt wird. In diesem Fall wird neben den Standardangaben von Name und der optionalen Beschreibung, bei der Art der Gruppenentscheidungsaufgabe “Suche & wähle eine Alternative“ ausgewählt.

Danach geht es weiter bei den spezifischen Einstellungen, die wie in den vorherigen Szenarien in einen allgemeinen und einen erweiterten Bereich aufgeteilt sind, wie aus der Abbildung 4.27 zu entnehmen ist.

Die vorhandenen Einstellungsmöglichkeiten ergeben sich aus der Kombination der bereits bekannten Einstellungen der beiden bisher beschriebenen Szenarien. Der Ersteller hat bei den allgemeinen Einstellungen die Option den Teilnehmern die Möglichkeit zu geben selbst Links oder Dateien hinzuzufügen. Weiters kann zwischen den verschiedenen vorhandenen Algorithmen zur Berechnung der Gruppenempfehlung gewählt werden.



(a) Allgemeine Einstellungen

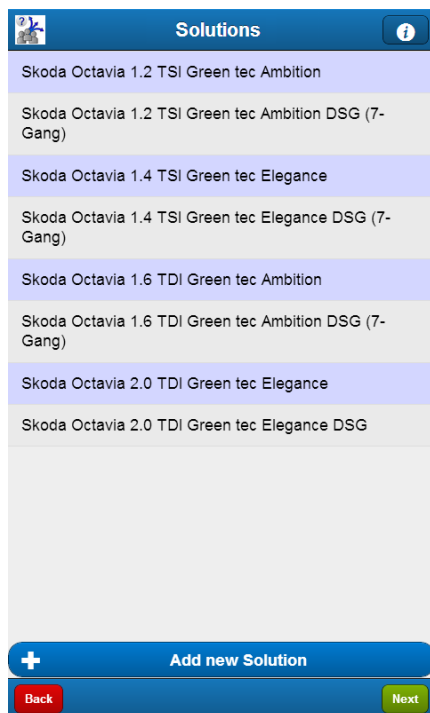
(b) Erweiterte Einstellungen

Abbildung 4.27: Spezifische Einstellungen des „Suche & wähle eine Alternative“ Szenarios

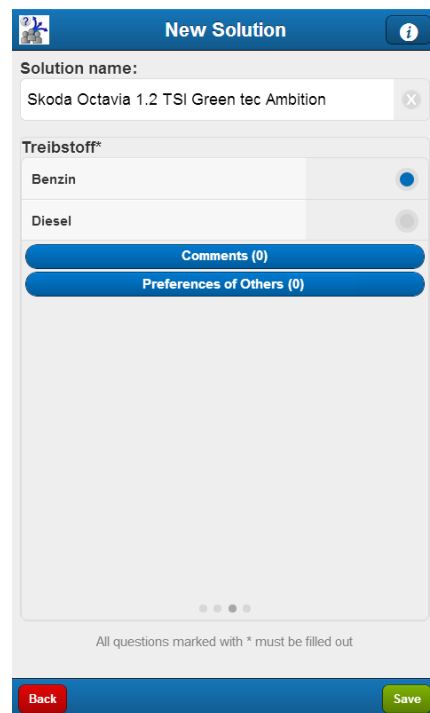
Die Art der Präferenzdarstellung der anderen Teilnehmer kann ebenfalls vom Ersteller angegeben werden, wobei im Beispiel, der Konfiguration eines neuen Autos, darauf verzichtet wurde. Ebenfalls wurde die Benutzerpräferenz auf "keine Präferenz" gesetzt. Die erweiterten Einstellungen beinhalten wieder die Optionen zur Kommunikation mit den Teilnehmern und bieten die Möglichkeit an, bei Abschluss der Gruppenentscheidungsfindung eine zusätzliche Erklärung für das finale Ergebnis anzubieten.

Nachdem alle Einstellungen getätigt wurden, wird mit der Erstellung der Fragenliste begonnen. Dies hat den gleichen Ablauf wie im Abschnitt 4.5 zum Szenario mit „Antworten finden“. Danach werden wie in Kapitel 4.6 mit dem Szenario „Wähle eine Alternative“ die Alternativen in einer Liste eingetragen, wie unter (a) von Abbildung 4.28 zu sehen ist.

Der Unterschied besteht dabei darin, dass jetzt nicht mehr nur eine *Messagebox* für den Namen zum Einsatz kommt, sondern ein Alternativeneditor, dessen Oberfläche unter (b) von Abbildung 4.28 dargestellt wird. Dieser Editor besitzt neben einem Feld für den Namen, auch eine *Carousel* mit den möglichen Antworten auf die Fragen die zuvor definiert wurden. Somit entspricht eine Alternative einer Art der Beantwortung aller Fragen, über die der Algorithmus die Gruppenempfehlung berechnen wird. Nachdem zwei oder mehr Alternativen eingetragen wurden, kann durch Auswahl des *Next*-Buttons mit der Finalisierung der Modellierung begonnen werden, da dies gleich abläuft wie in allen Arten der Gruppenentscheidungsaufgaben und bereits besprochen wurde, wird hier auf eine erneute Erörterung verzichtet und mit dem Kapitel zur Teilnahme fortgesetzt.



(a) Alternativenliste

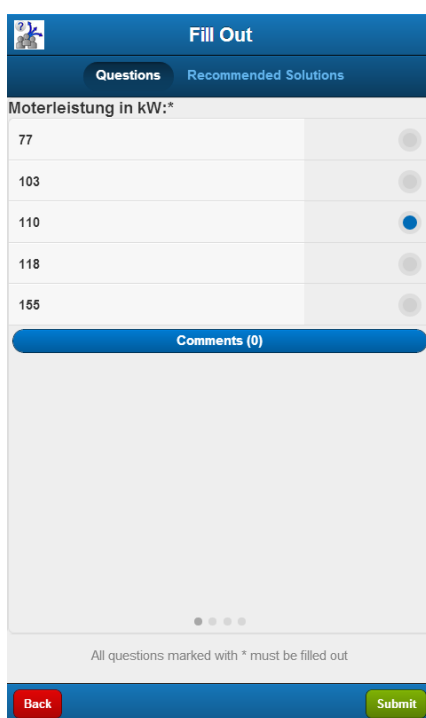


(b) Editor für die Alternativen

Abbildung 4.28: Modellierung der Alternativen des „Suche & wähle eine Alternative“ Szenarios

4.7.2 Teilnahme

Nachdem die Teilnehmer vom Administrator zu der Gruppenentscheidungsfindung hinzugefügt wurden, wie in dem Kapitel 4.1 über die Verwaltung der Gruppenentscheidungsaufgaben beschrieben, können die Teilnehmer wie bisher auch über die Detailansicht der Aufgabe mit der Präferenzabgabe über den *Fill Out*-Button beginnen. Der Ablauf der Gruppenentscheidungsfindung ist dabei wieder in zwei Phasen unterteilt.



Stufe 1: Präferenzabgabe mittels Fragenliste



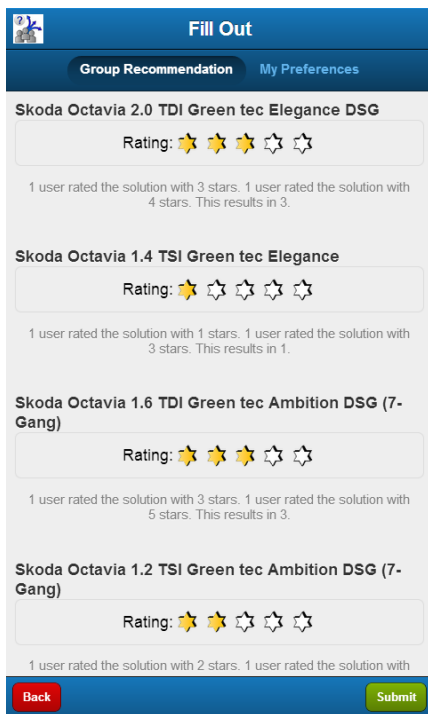
Stufe 2: Präferenzabgabe anhand der empfohlenen Alternativen

Abbildung 4.29: Phase 1 des „Suche & wähle eine Alternative“ Szenarios

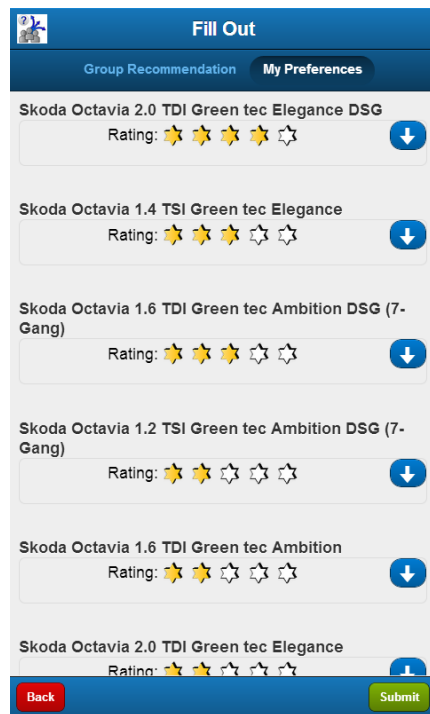
Die Präferenzenabgabe in der ersten Phase erfolgt in zwei Stufen, die beide in Abbildung 4.29 dargestellt sind. Zuerst werden die Präferenzen der Teilnehmer anhand der während der Modellierung erstellten Fragenliste aufgenommen, dies geschieht gleich wie bei der Teilnahme an einem „Antworten finden“ Szenario, nachzulesen in Kapitel 4.5.2. Der Server berechnet daraufhin Empfehlungen, die in der zweiten Stufe die angezeigten Alternativen auf

die für den Benutzer relevanten reduziert. Die Alternativenliste wird dabei mit absteigender Sortierung ihrer Bewertung angezeigt und jedem Eintrag seine Bewertung angefügt. Jetzt verhält es sich gleich wie bei der Teilnahme an einem „Wähle eine Alternative“ Szenario, dass bereits in Kapitel 4.6.2 besprochen wurde. Die Präferenzen werden anhand der bekannten 5-Sterne-Skala abgegeben.

Nachdem der Administrator den Teilnehmern genügend Zeit für die Präferenzabgabe gegeben hat, lässt er die Gruppenentscheidungsfindung in die nächste Phase fortschreiten. In Phase 2 der Gruppenentscheidungsaufgabe verschwindet die Fragenliste aus Phase 1 und wird durch die Ansicht der Gruppenempfehlung ersetzt. Bei jedem Eintrag in der Alternativenliste wird eine Beschreibung angeboten, welche die Bewertung erläutern soll. Wie üblich können die Teilnehmer ihre Präferenzabgabe überarbeiten. Die Abbildung 4.30 zeigt die Oberfläche bei der Teilnahme in der zweiten Phase.



(a) Gruppenempfehlung

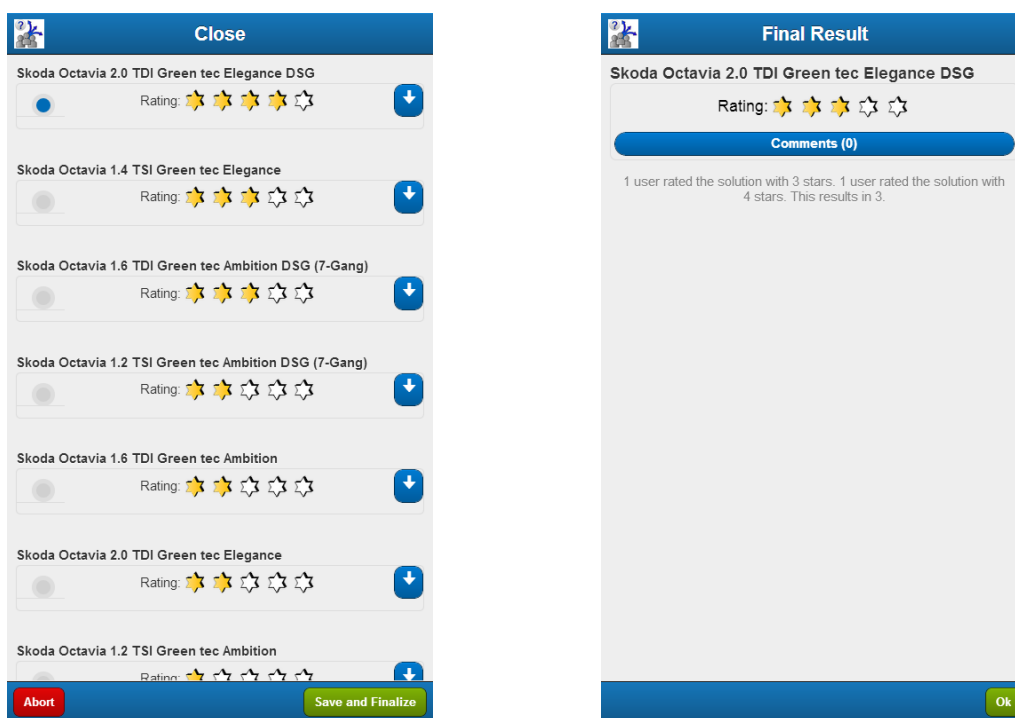


(b) Präferenzanpassung

Abbildung 4.30: Phase 2 des „Suche & wähle eine Alternative“ Szenarios

4.7.3 Finalisierung

Dem Administrator obliegt es abermals die Finalisierung einzuleiten. Dies geschieht gleich wie beim Szenario „Wähle eine Alternative“, bei dem die Alternativen inklusive ihrer Bewertung angezeigt werden (siehe (a) von Abbildung 4.31) und der Administrator eine der Alternativen als finales Ergebnis der Gruppenentscheidungsaufgabe festlegt. Dafür wählt er die zu dem Eintrag gehörende *Checkbox* aus und betätigt zum Abschluss den *Save and Finalize*-Button. Das finale Ergebnis, zu sehen in (b) von Abbildung 4.31, wird den Teilnehmern wie gewohnt präsentiert und die Gruppenentscheidungsfindung ist damit beendet.



Finalisierung

Finales Ergebnis

Abbildung 4.31: Abschluss des „Suche & wähle eine Alternative“ Szenarios

4.8 Einen Termin finden

Dieser Abschnitt befasst sich mit dem Terminfindungsszenario, welches es den Teilnehmern ermöglichen soll, einen passenden gemeinsamen Termin zu finden. Dabei obliegt es nicht nur einer Person passende Termine vorzuschlagen und der Rest der Gruppe gibt nur bekannt ob diese Termine passend sind oder nicht, sondern jeder Teilnehmer der Gruppenentscheidungsaufgabe gibt seinen individuellen zeitlichen Rahmen bekannt und das System präsentiert unterschiedliche Möglichkeiten für Termine aus denen der Administrator schlussendlich eine Auswahl treffen soll. Weiters besteht die Möglichkeit diesen individuellen Zeiteinteilungen Prioritäten zu geben, wodurch der verwendete Algorithmus weitere Parameter bekommt um eine möglichst optimale Zeitspanne für das Treffen zu finden. Als Beispiel für das Terminfindungsszenario wird hier das Finden eines passenden Termins für ein Teammeeting, so wie es in jeder Firma vorkommen könnte, verwendet. Abermals ist der Aufbau des Abschnitts in Modellierung, Teilnahme und Finalisierung unterteilt.

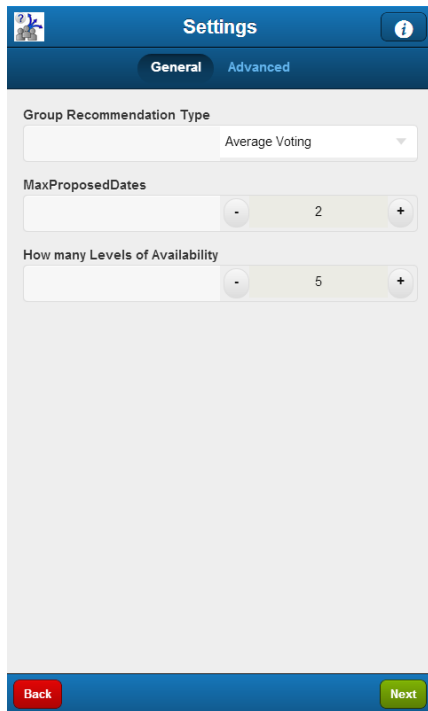
4.8.1 Modellierung

Am Beginn der Modellierung des „Einen Termin finden“-Szenarios steht wieder das Basis-Formular (siehe Abbildung 4.13 in Kapitel 4.5.1), bei dem wie in den drei vorausgegangenen *Choice Decision*-Szenarien der Name der Gruppenentscheidungsaufgabe und falls gewünscht eine Beschreibung dieser anzugeben ist. Um ein Terminfindungsszenario zu erstellen, muss hier bei der Art der Gruppenentscheidung „Einen Termin finden“ ausgewählt werden.

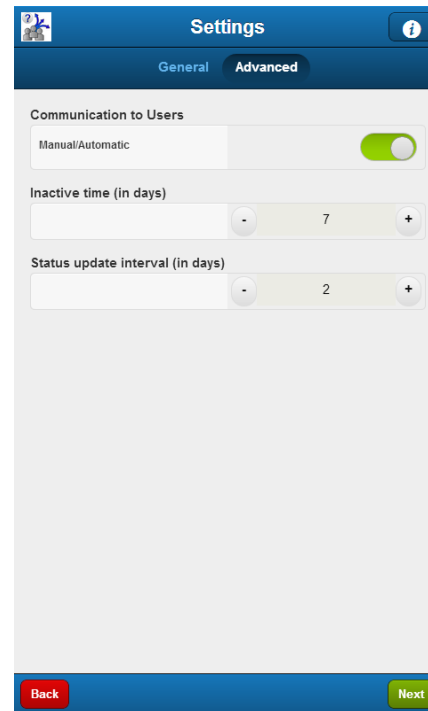
Als Nächstes kommen die spezifischen Einstellungen, die sich von den bisher bekannten Einstellungen erheblich unterscheiden. Diese sind auch wieder in einen allgemeinen und einen erweiterten Bereich unterteilt, wie in Abbildung 4.32 dargestellt.

Die erste allgemeine Einstellungsoption beinhaltet die Auswahl der Art des verwendeten Gruppenentscheidungsalgorithmus. Diese Option ist aber bereits aus den vorherigen Szenarien bekannt. Interessanter wird es da bei der nächsten Einstellungsmöglichkeit, bei der es um die maximale Anzahl der vorgeschlagenen Termine geht. Der Standardwert liegt hier bei drei. Die dritte und letzte Option der allgemeinen Einstellungen gibt an wie viele Level für die

Auswahl eines Feldes bei den Teilnehmern verfügbar sind. Dieser Wert kann von zwei bis fünf gesetzt werden und gibt im später noch gezeigten Kalender die Möglichkeit Level von „Geht gar nicht“ bis „Passt mir perfekt“ den einzelnen Felder zuzuweisen.



(a) Allgemeine Einstellungen



(b) Erweiterte Einstellungen

Abbildung 4.32: Spezifische Einstellungen des „Einen Termin finden“ Szenarios

Bei den erweiterten Einstellungen scheint wieder die bereits bekannte Option zur Kommunikation mit den Teilnehmern auf. Dies muss an dieser Stelle nicht erneut erörtert werden, sondern kann in den vorherigen Kapiteln nachgelesen werden.

Die weitere Modellierung des „Einen Termin finden“ Szenarios verhält sich im Vergleich zu den Vorgängern ein wenig anders. Dabei wird die Zeiteinteilung (Abbildung 4.33) bestehend aus *Zeitraster* und *Tagesauswahl* festgelegt, zwischen denen wie bisher auch mittels einer *Tabbar* navigiert werden kann.

Das Start- und End-Datum des Zeitrasters wird mittels eines *Date-Pickers* eingetragen, wobei die Daten nicht in der Vergangenheit liegen dürfen. Ein *Slider* dient zur Einschränkung der Uhrzeiten an diesem Tag, dabei gibt der linke *Thumb* die Start-Uhrzeit und der rechte *Thumb* die End-Uhrzeit an den Tagen an. Die Länge der Einheiten im Kalender wird im nächsten Feld angegeben, dabei gibt es die Auswahlmöglichkeit von ½, 1 und 2 Stunden. Zuletzt steht noch die Frage nach der Dauer der Meetings, dies kann mit einer halben bis fünf Stunden angegeben werden.

The image shows two screenshots of a mobile application interface for scheduling a meeting. Both screenshots are titled "Timing" and have a blue header with a user profile icon and an information icon. The first screenshot, labeled (a), shows the "Time Period" tab selected. It contains fields for "Start Date" (02/10/2014) and "End Date" (02/16/2014). Below these is a "Hour: Start - End" section with a slider set to 7:00 - 20:00. Further down are "Units" (01:00) and "Duration of Meeting" (02:00). The second screenshot, labeled (b), shows the "Day Options" tab selected. It lists the days of the week from Monday to Sunday, each with a blue checkmark indicating selection. Both screenshots have a "Back" button in a red box and a "Next" button in a green box at the bottom.

(a) Zeitraster

(b) Tagesauswahl

Abbildung 4.33: Zeiteinteilung des „Einen Termin finden“ Szenarios

Nach Validierung der Eingabe kann mit der Finalisierung der Modellierung fortgesetzt werden, die gleich wie am Ende von Kapitel 4.5.1 abläuft.

4.8.2 Teilnahme

Nachdem der Ersteller der Terminfindungsaufgabe die Teilnehmer hinzugefügt hat, können diese wieder über die Detailansicht der Gruppenentscheidungsaufgabe und der Auswahl des *Fill Out*-Buttons ihre Präferenzen abgeben. Dabei ist der Ablauf der Teilnahme gleich wie in den vorherigen Szenarien in zwei Phasen aufgeteilt, wobei in der ersten Phase abermals jeder Teilnehmer auf sich selbst gestellt ist und somit ohne Einflussnahme der anderen Teilnehmer wirken kann und erst in der zweiten Phase die vorläufige Gruppenempfehlung angezeigt wird.

Die Oberfläche zu Phase 1 von „Einen Termin finden“ ist in Abbildung 4.34 dargestellt. Sie besteht aus einem Kalender in Tabellen-Form, bei dem die Spalten die Tage und die Zeilen die Uhrzeit an diesen Tagen angeben. Die Navigation kann stufenlos mittels Wischbewegung vollführt werden und ermöglicht so einfaches Scrollen innerhalb des Kalenders.

Fill Out				
Time	Monday 10.02.2014	Tuesday 11.02.2014	Wednesday 12.02.2014	Thursday 13.02.2014
7:00 - 8:00	works		Very High	
8:00 - 9:00	High	Low	Very High	
9:00 - 10:00	Very High	works	Very High	
10:00 - 11:00	Very High	High	Very High	
11:00 - 12:00	Very High	Very High	Very High	
12:00 - 13:00	High	Very High	High	High
13:00 - 14:00		Very High	works	works
14:00 - 15:00		Very High	Low	works
15:00 - 16:00		Very High		works
16:00 - 17:00		Very High		works
17:00 - 18:00		Very High		works
18:00 - 19:00		High		
19:00 - 20:00		works		

■ Very High
 ■ High
 ■ works
 ■ Low
 ■ Very Low

Back
Submit

Abbildung 4.34: Phase 1 des “Einen Termin finden“ Szenarios

Durch Auswahl eines der Felder in dem Kalender kann das Level dieses Feldes verändert werden. Im Beispiel dieses Abschnittes gibt es fünf verschiedene Level, die auch aus der Legende am unteren Rand der Oberfläche ersichtlich sind. Den Teilnehmern stehen damit alle Mittel zur Verfügung um ihre Präferenzen über den während der Modellierung gewählten Zeitraum einzugeben. Nachdem im Kalender alle Präferenzen festgelegt wurden, werden die Daten mittels Auswahl des *Submit*-Buttons an den Server übermittelt.

Nachdem genug Teilnehmer ihre Präferenzen abgegeben haben sowie eine angemessene Zeitspanne vergangen ist, steht es dem Administrator der Gruppenentscheidungsaufgabe frei in die nächste Phase der Gruppenentscheidungsfindung fortzuschreiten. Dafür wählt er in der Detailansicht, wie bereits im Kapitel 4.1 näher erklärt, den *Advance*-Button aus und ermöglicht damit den Eintritt in die zweite Phase der Gruppenentscheidungsfindung.

Time	Monday 10.02.2014	Tuesday 11.02.2014	Wednesday 12.02.2014	Thursday 13.02.2014
7:00 - 8:00	works	Very Low	Very High	Very Low
8:00 - 9:00	High	Low	Very High	Very Low
9:00 - 10:00	Very High	works	Very High	Very Low
10:00 - 11:00	Very High	High	Very High	Very Low
11:00 - 12:00	Very High	Very High	Very High	Very Low
12:00 - 13:00	High	Very High	High	High
13:00 - 14:00	Very Low	Very High	works	works
14:00 - 15:00	Very Low	Very High	Low	works
15:00 - 16:00	Very Low	Very High	Very Low	works
16:00 - 17:00	Very Low	Very High	Very Low	works
17:00 - 18:00	Very Low	Very High	Very Low	works
18:00 - 19:00	Very Low	High	Very Low	Very Low
19:00 - 20:00	Very Low	works	Very Low	Very Low

(a) Gruppenempfehlung

Time	Monday 10.02.2014	Tuesday 11.02.2014	Wednesday 12.02.2014	Thursday 13.02.2014
7:00 - 8:00				
8:00 - 9:00				
9:00 - 10:00		2		
10:00 - 11:00			1	
11:00 - 12:00				
12:00 - 13:00				
13:00 - 14:00				
14:00 - 15:00				
15:00 - 16:00				
16:00 - 17:00				
17:00 - 18:00				
18:00 - 19:00				
19:00 - 20:00				

(b) Präferenzanpassung

Abbildung 4.35: Phase 2 des „Einen Termin finden“ Szenarios

In Phase 2 der Gruppenentscheidungsaufgabe (Abbildung 4.35) bekommen die Teilnehmer durch einen zweiten Bereich in einem weiteren Kalender die von dem Algorithmus berechneten Empfehlungen präsentiert. Die Nummern geben dabei aufsteigend die Reihung der Empfehlung an. Mittels einer bereits bekannten *Tabbar* wird dabei zwischen den Bereichen navigiert. Den Teilnehmern steht es frei ihre eigenen Präferenzen anzupassen und diese wieder dem Server zu übermitteln.

4.8.3 Finalisierung

Dem Administrator obliegt die Aufgabe die Finalisierung mithilfe des unter (a) in Abbildung 4.36 dargestellten Formulars einzuleiten.

Time	Monday 10.02.2014	Tuesday 11.02.2014	Wednesday 12.02.2014	Thursday 13.02.2014
7:00 - 8:00				
8:00 - 9:00				
9:00 - 10:00		2		
10:00 - 11:00			1	
11:00 - 12:00				
12:00 - 13:00				
13:00 - 14:00				
14:00 - 15:00				
15:00 - 16:00				
16:00 - 17:00				
17:00 - 18:00				
18:00 - 19:00				
19:00 - 20:00				

(a) Finalisierung

Time	Monday 10.02.2014	Tuesday 11.02.2014	Wednesday 12.02.2014	Thursday 13.02.2014
7:00 - 8:00				
8:00 - 9:00				
9:00 - 10:00				
10:00 - 11:00				
11:00 - 12:00				
12:00 - 13:00				
13:00 - 14:00				
14:00 - 15:00				
15:00 - 16:00				
16:00 - 17:00				
17:00 - 18:00				
18:00 - 19:00				
19:00 - 20:00				

(b) Finales Ergebnis

Abbildung 4.36: Abschluss des „Einen Termin finden“ Szenarios

Dabei wird wieder ein Kalender mit den abschließenden Gruppenempfehlungen angezeigt und der Administrator wählt durch Auswahl eines der empfohlenen Felder das finale Ergebnis aus. Das Datum und die Uhrzeit werden dann in einer eigenen Zeile über dem Kalender angezeigt und die Finalisierung kann durch Betätigung des *Save and Finalize*-Buttons abgeschlossen werden.

Das finale Ergebnis, das unter (b) in Abbildung 4.36 abgebildet ist, wird den Benutzern wie üblich mittels des *Details*-Buttons der Detailansicht der Gruppenentscheidungsaufgabe angezeigt und beinhaltet nur mehr den leeren Kalender mit dem ausgewählten, gefärbten Termin.

4.9 Anforderungen priorisieren

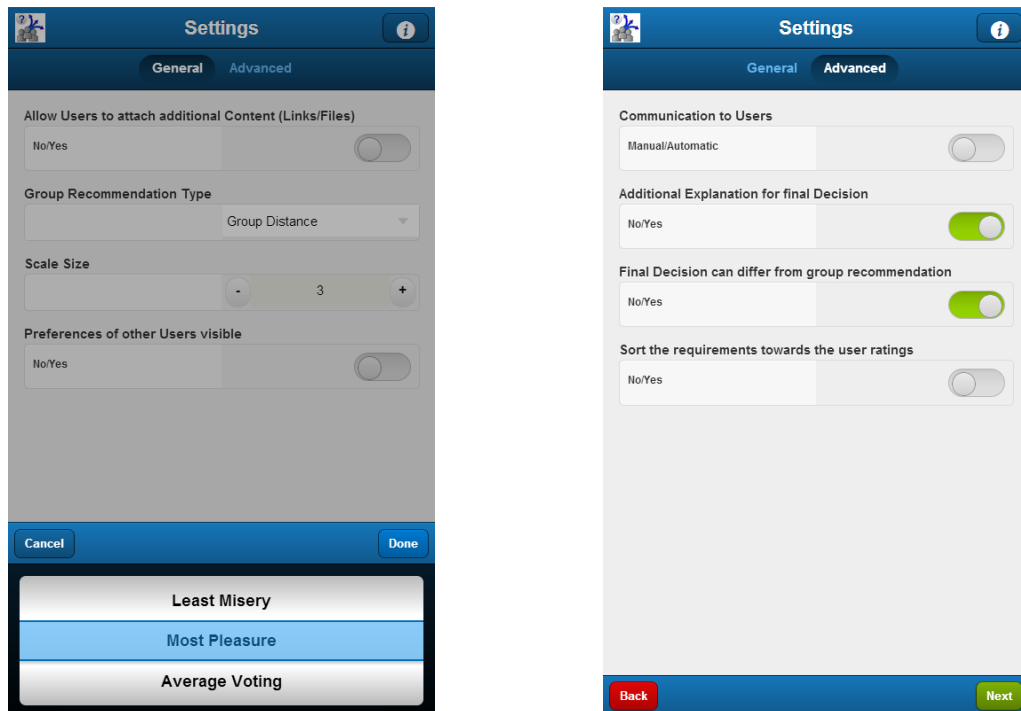
Dieser Abschnitt beschäftigt sich mit der Anforderungspriorisierung, die in der Softwareentwicklung ein heikles Thema darstellen kann [Felfernig, Zehentner, Ninaus, Grabner, Maalej, Pagano, Weninger und Reinfrank, 2011]. Dabei werden Anforderungen aufgestellt, die dann von einer Gruppe nach ihrer Priorität geordnet werden sollen. In diesem Abschnitt wird dazu anhand des Beispiels einer Softwareentwicklung der Ablauf innerhalb der Applikation erläutert.

4.9.1 Modellierung

Auch beim letzten Szenario steht am Anfang wieder das Basis-Formular. Dieses Mal wird neben den Standardeingaben des Namens und der wahlfreien Beschreibung, bei der Art der Gruppenentscheidungsaufgabe „Anforderungen priorisieren“ ausgewählt.

Danach geht es wie gehabt mit den spezifischen Einstellungen dieses Szenarios weiter, in denen sich bereits bekannte Einstellungsoptionen wiederfinden, aber auch ein paar neue Punkte vorkommen, wie aus der Abbildung 4.37 zu entnehmen ist.

Zu den altbekannten allgemeinen Einstellungen gehört das Zulassen von zusätzlichen Inhalten der Teilnehmer in Form von Links oder Dateien, sowie die Auswahl des zu verwendeten Gruppenentscheidungsalgorithmus mittels *Picker*, der in den allgemeinen Einstellungen der Abbildung 4.37 gerade aktiv ist. Ebenfalls bekannt ist die Frage nach der Sichtbarkeit der Präferenzen der anderen Teilnehmer und der damit einhergehenden Auswahl der Darstellungsform. Zu den neuen Einstellungsoptionen gehört die Größe der Skala, die zwischen 2 und 10 angegeben werden kann und mittels Spinner eingestellt wird. Die Größe der Skala wird später die Darstellungsform der Präferenzabgabe beeinflussen.



(a) Allgemeine Einstellungen

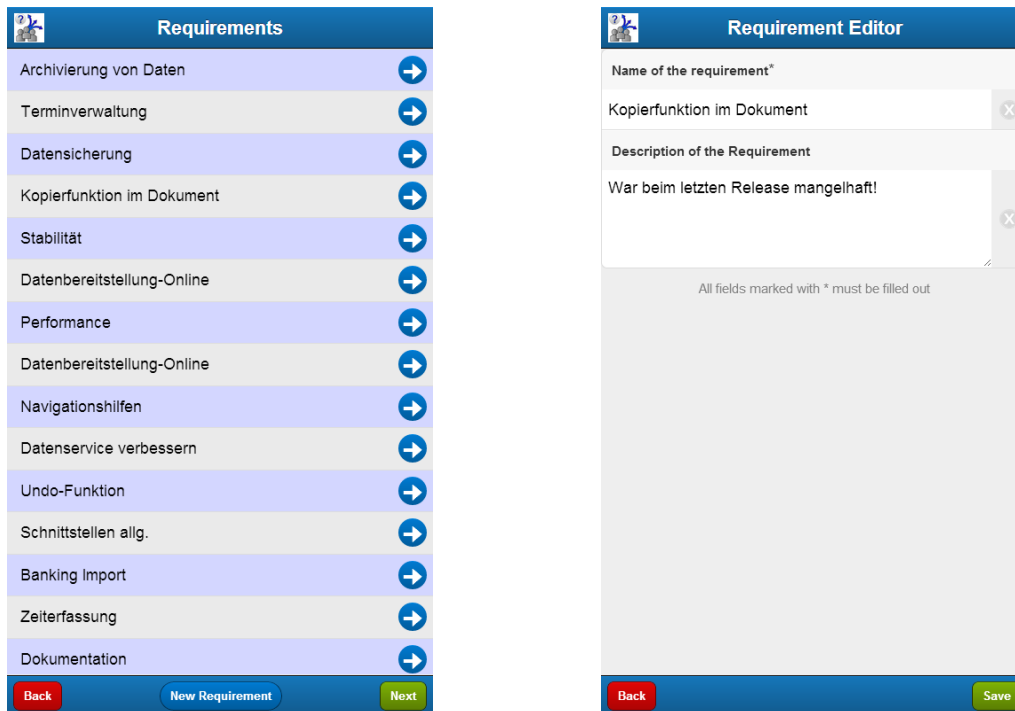
(b) Erweiterte Einstellungen

Abbildung 4.37: Spezifische Einstellungen des „Anforderungen priorisieren“ Szenarios

Zum erweiterten Bereich der Einstellungen gehört wieder die Kommunikation mit den Teilnehmern, wo es darum geht ob sie *automatisch* oder *manuell* abgehalten wird. Auch gibt es wieder die Möglichkeit, dass der Administrator ein finales Ergebnis unabhängig von der Empfehlung des Algorithmus beschließen kann. Sowie der Möglichkeit einer zusätzlichen Erklärung in Form eines kurzen Textes, der zum Einsatz kommen sollte, falls die vorherige Option verwendet wird. Zu den Neuerungen in diesem Bereich zählt die Sortierung der Anforderungen hinsichtlich der Benutzerbewertung.

Nach den spezifischen Einstellungen geht es weiter mit der Aufstellung der Anforderungsliste, die unter (a) in Abbildung 4.38 abgebildet ist. Dafür kommt der in (b) dargestellte spezielle Editor zum Einsatz, der für das Erstellen neuer Anforderungen und das Editieren bereits vorhandener Anforderungen verwendet wird.

Durch das betätigen des *New Requirement*-Buttons, wird der Editor mit einer neuen Anforderung geöffnet. Er besteht aus dem Pflichtfeld mit dem Namen der Anforderung und einem Textfeld zur optionalen Eingabe der Beschreibung. Nachdem der *Save*-Button ausgewählt



(a) Anforderungsliste

(b) Anforderungseditor

Abbildung 4.38: Modellierung der Anforderungen des „Anforderungen priorisieren“ Szenarios

wurde, wird die Eingabe, wie bisher auch, validiert und in der Liste angezeigt. Um einen Eintrag in der Liste zu editieren, genügt es diesen auszuwählen.

Nach der Eingabe von mindestens zwei Anforderungen kann mit der Finalisierung fortgesetzt werden, diese läuft gleich ab wie am Ende des Kapitels 4.5.1 beschrieben.

4.9.2 Teilnahme

Nach Abschluss der Modellierung kann der Administrator die Teilnehmer zur Gruppenentscheidungsaufgabe hinzufügen. Danach können diese über den *Fill Out*-Button der Detailansicht in der ersten Phase der Teilnahme ihre Präferenzen abgeben. Abbildung 4.39 zeigt die Liste mit den Anforderungen, wobei abhängig von der während der Modellierung eingestellten Größe der Skala zwischen 2 und 10 Sterne angezeigt werden. Im angegebenen Beispiel zur Softwareentwicklung wurde eine Größe von 5 gewählt. Jeder Eintrag der Anforderungsliste enthält einen Button für ein *Dropdown*-Menü des Feldes, wodurch dieses die Beschreibung der Anforderung preis gibt. Nachdem der Benutzer das Formular ausgefüllt hat und dieses durch den *Submit*-Button an den Server gesendet wurde, kann er darauf warten, dass der Administrator die Gruppenentscheidung auf Phase 2 fortschreiten lässt.



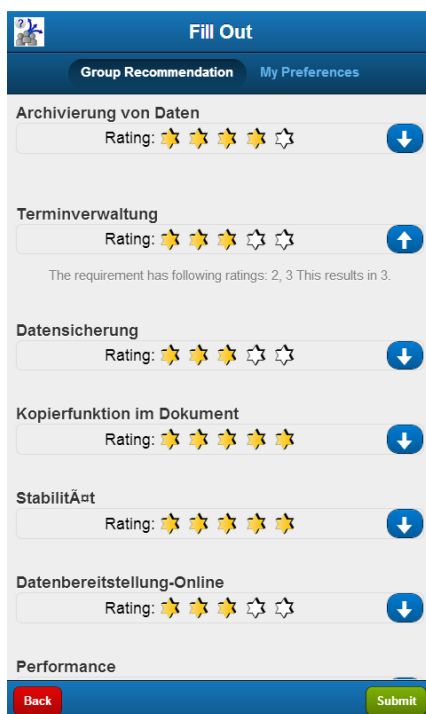
The screenshot shows a mobile application interface titled "Fill Out". It displays a list of requirements for prioritization. Each requirement is presented in a row with a text label, a "Rating:" field, and a dropdown arrow button. The requirements listed are:

- Datenservice verbessern (Rating: 5 stars)
- Undo-Funktion (Rating: 5 stars)
- Schnittstellen allg. (Rating: 2 stars)
- Banking Import (Rating: 5 stars)
- Zeiterfassung (Rating: 3 stars)
- Dokumentation (Rating: 4 stars)
- Hilfe (Rating: 4 stars)

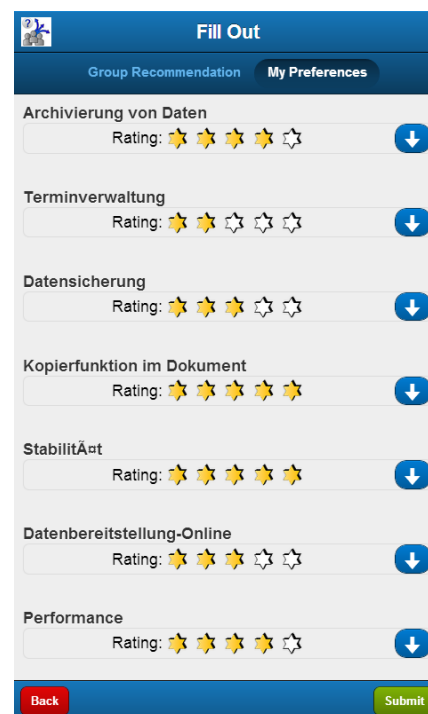
At the bottom of the form, there are two buttons: a red "Back" button and a green "Submit" button.

Abbildung 4.39: Phase 1 des „Anforderungen priorisieren“ Szenarios

In Phase 2 der Gruppenentscheidungsfindung wird ein *Gruppenempfehlungs-Tab* der *Tabbar* hinzugefügt, mit der dessen Ansicht aktiviert werden kann. Die Gruppenempfehlungsansicht präsentiert die vorläufigen Ergebnisse des Algorithmus, wie in Abbildung 4.40 dargestellt wird. Hier gibt es wie bei der Präferenzabgabe ebenfalls eine *Dropdown-Fähigkeit*, wodurch die Erklärung über die Berechnung der Bewertung der jeweiligen Anforderung angezeigt wird. Die Präferenzen der Teilnehmer können wieder nach ihren Vorstellungen angepasst und an den Server übermittelt werden.



(a) Gruppenempfehlung



(b) Präferenzanpassung

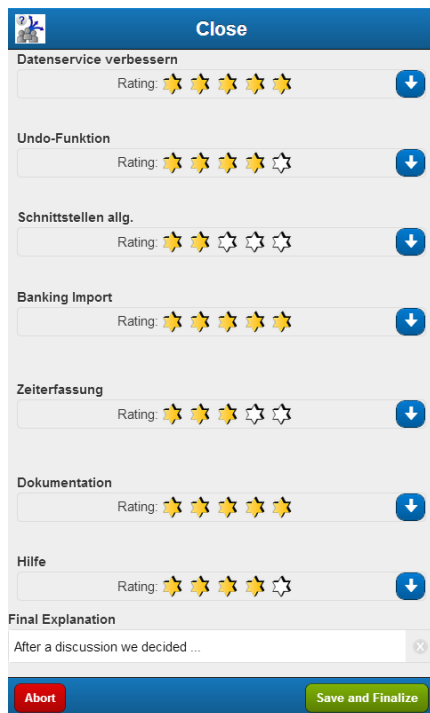
Abbildung 4.40: Phase 2 des „Anforderungen priorisieren“ Szenarios

4.9.3 Finalisierung

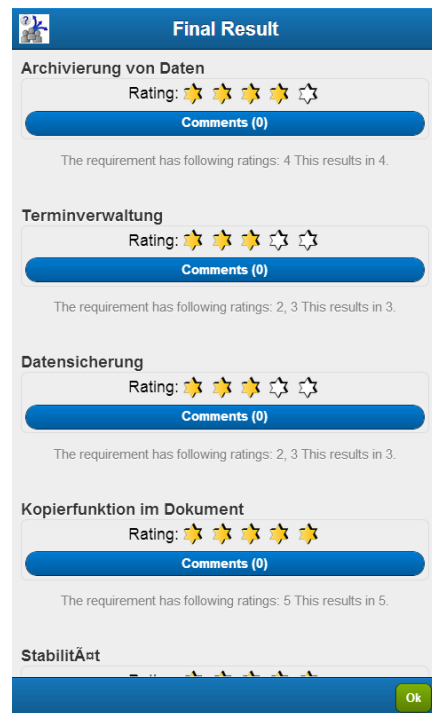
Der Finalisierungsprozess schließt die Gruppenentscheidung ab, er wird vom Administrator initiiert und legt im konkreten Beispielszenario die Reihung der Anforderungen fest. Unter (a) in Abbildung 4.41 wird die Darstellung der Anforderungen mit der vom Algorithmus

berechneten Bewertung angezeigt. Da sich der Ersteller während des Modellierungsprozesses die Möglichkeit offen gelassen hat die vorgeschlagene Empfehlung anpassen zu können, kann er dies jetzt machen. Zusätzlich wurde bei den spezifischen Einstellungen der Zusatz für eine Erklärung aktiviert, weshalb am unteren Ende der Ansicht ein Textfeld für genau jenes angezeigt wird. Darin kann der Administrator eine Erklärung abgeben über die von ihm getroffene Entscheidung. Dabei spielt nicht nur die Bewertung sowie deren Zustandekommen eine Rolle, sondern möglicherweise auch abgegebene Kommentare der Benutzer. Hat der Administrator alles zu seiner Zufriedenheit angepasst, kann er durch das Betätigen des *Save and Finalize*-Buttons die Finalisierung abschließen.

Nach Abschluss verschwindet die Gruppenentscheidungsaufgabe aus den aktuellen Entscheidungen der Hauptseite und wandert zu den geschlossenen Entscheidungen. Dort können die Teilnehmer das finale Ergebnis begutachten, das ihnen hoffentlich den Aufschluss gibt den sie gesucht haben. Unter (b) in Abbildung 4.41 wird die Ansicht dieses Resultats angezeigt.



(a) Finalisierung



(b) Finales Ergebnis

Abbildung 4.41: Abschluss des „Anforderungen priorisieren“ Szenarios

4.10 Evaluierung

Die Evaluierung der grafischen Benutzeroberfläche wurde mit einer Testgruppe von 20 Personen durchgeführt. In der ersten Phase wurden die Testpersonen dazu angehalten sich mit dem System selbstständig vertraut zu machen und eigene Szenarien anzulegen. In der zweiten Phase wurden dann Aufgaben verteilt, die von den Testpersonen durchgeführt werden sollten. Nach der Erfüllung der Aufgaben wurde ein Fragebogen der auf den System Usability Scale (SUS) basiert vorgelegt und die Testpersonen durften die Applikation bewerten.

System Usability Scale bietet eine quantitative Methode zur Messung der Bedienbarkeit von Interfaces und umfasst einen Fragebogen mit 10 Fragen mit einer fünfteiligen Antwortskala im Likert-Format [Likert, 1932] von „Stimme gar nicht zu“ bis „Stimme voll zu“. Ursprünglich wurde SUS von John Brook 1986 entwickelt und ermöglicht die Bewertung von unterschiedlichsten Interfaces. Seither wurde SUS in vielen Artikeln und Puplicationen referenziert. [Tullis und Albert, 2008]

WeDecideMobile wurde als einfach zu bedienen eingestuft und eine große Zahl der Testpersonen empfand die verschiedenen Funktionen als gut integriert. Abbildung 4.42 zeigt das Ergebnis der Evaluierung. Die Applikation bekam dabei die durchschnittliche SUS-Auswertung von 83,75% und kann sich somit als gutes System bezeichnen [Bangor, Kortum und Miller, 2008].

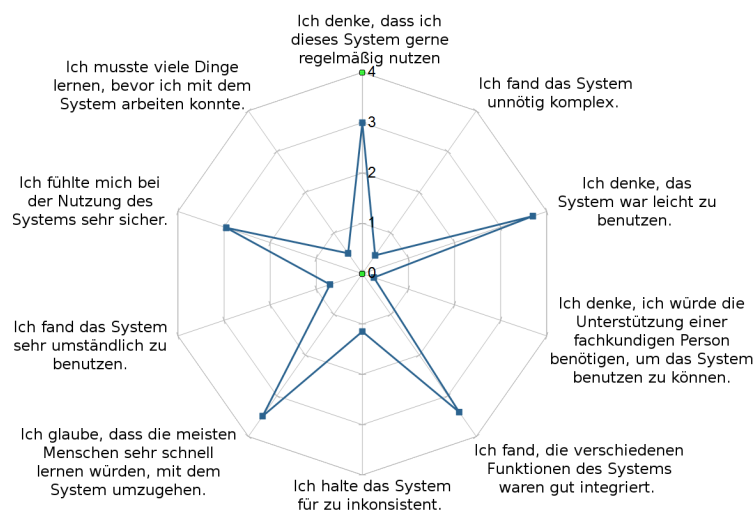


Abbildung 4.42: Evaluierung mittels System Usability Scale

5 Zusammenfassung

Diese Arbeit beschäftigt sich damit Gruppenempfehlungssysteme auf mobilen Geräten zugänglich zu machen. Dabei liegt der Fokus neben der Präsentation, der Benutzerfreundlichkeit und dem intuitiven User Interface (UI) der Applikation, auf der großflächigen Abdeckung des Smartphone- und Tabletmarkts, mit den vielen unterschiedlichen Betriebssystemen. Ebenso soll ein möglichst großer Bereich der Gruppenempfehlungsszenarien abgedeckt werden. Ein Nebenziel ist die Umsetzung des Projekts mit Hilfe testgetriebener Softwareentwicklung (engl. Test Driven Development).

Dies wurde, wie in den folgenden Absätzen erläutert, durch die entwickelte Applikation umgesetzt.

Der Kernpunkte der Arbeit war die großflächige Abdeckung des Smartphone- und Tabletmarktes. Dafür wurde zu Beginn des Projekts eine Recherche zu Vor- und Nachteile von Native, Web- oder Hybrid-Applikationen durchgeführt. Die Hybrid-Applikation eignete sich am besten zur Umsetzung des Projekts, da durch die plattformübergreifende Entwicklung nahezu alle mobilen Betriebssysteme bedient werden können, ohne für jedes individuelle Betriebssystem eine eigene Applikation entwickeln zu müssen. In diesem Zusammenhang fiel die Wahl auf das Javascript Framework *Sencha Touch*, dass sich durch seine Flexibilität, der großen Auswahl an UI-Elementen und der eingesetzt MVC-Architektur hervortat. Das grafische Benutzerinterface (engl. Graphical User Interface) ähnelt den nativen Applikationen auf einem *iOS* Betriebssystem und ist benutzerfreundlich mit einer intuitiven Menüführung. Die Performance von Hybrid-Applikationen mit *Sencha Touch* konnte leider mit denen der nativen Applikationen nicht mithalten. Auch wurden während der Entwicklung etliche wichtige Updates für *Sencha Touch* veröffentlicht, die zu Problemen mit dem vorhandenen Code führten.

Da der Fokus der Arbeit auf der Zugänglichkeit für mobile Geräte und nicht der Neuentwicklung eines Gruppenempfehlungssystems lag, wurde zu Beginn des Projekts eine Evaluierung bereits vorhandener Gruppenempfehlungssysteme durchgeführt. Diese ergab, dass ein domänenunabhängiges Gruppenempfehlungssystem, durch die Abdeckung unterschiedlichster Empfehlungsszenarien, den Zielen dieser Arbeit am meisten dienlich ist. Deshalb wurde das domänenunabhängige Gruppenempfehlungssystem *WeDecide* [Stettinger, Ninaus, Jeran, Reinfrank und Reiterer, 2013] als Grundgerüst eingesetzt und darauf eine Schnittstelle für die mobile Applikation entwickelt.

Die Entwicklung der hybriden Applikation wurde mit Hilfe von verhaltensgetriebener Softwareentwicklung (engl. Behavior Driven Development) unter Verwendung des Frameworks *Jasmine* umgesetzt, womit auch das Nebenziel erreicht wurde. Die Applikation wurde dabei in einem *Google Chrome Browser* geöffnet und darin die Tests durchgeführt, da das Remote Debugging (Schritt-für-Schritt Debuggen) innerhalb eines mobilen Betriebssystems, wie *Android*, durch die verwendete Web-View nicht funktioniert.

Auch wenn die Umsetzung des Projekts erfolgreich war, gibt es immer noch Platz für Verbesserungen, die im nächsten Kapitel *Offene Forschungsfragen* skizziert werden.

6 Offene Forschungsfragen

Die aktuelle Applikation hat es geschafft das Gruppenempfehlungssystem *WeDecide* für mobile Geräte zugänglich zu machen. Jedoch gibt es noch einiges, was in zukünftigen Projekten verbessert oder erweitert werden könnte. Nachstehend werden ein paar der möglichen Ansätze aufgezeigt.

Wie bereits in der Zusammenfassung angemerkt, leidet die Applikation durch ihre Web-Technologien und der momentan Version von *Sencha Touch* unter Performance Problemen. Dies könnte durch zukünftige Upgrades des Frameworks oder durch Verbesserungen im Source Code, zum Beispiel durch die Reduktion oder den Verzicht der Suche im DOM-Tree, erreicht werden.

Die momentane Applikation benötigt bei jedem ihrer Schritte eine Verbindung zum Server und somit auch zum Internet. Eine Möglichkeit wäre ein Offline-Modus, bei dem die Benutzer ihre mobilen Geräte durch öffnen eines W-Lan Knotenpunkts oder der Verwendung von Bluetooth miteinander verbinden. Dabei würde einer der Clients die Rolle des Servers übernehmen und die Verwaltung übernehmen. Stellt die Applikation wieder eine Internetverbindung fest, kann sie die Daten mit dem Server abgleichen.

Bei mobilen Applikationen die komplexere Themen behandeln, besteht öfters Unklarheit in der Bedienung oder im Ablauf unterschiedlicher Funktionen. Auch wenn die aktuelle Version intuitiv einsetzbar ist, könnte ein Tutorial in Form von Videos auftretende Fragen beantworten oder den Ablauf erklären. Eine Möglichkeit wäre ein Button mit einem Info-Icon in der rechten oberen Ecke der Seiten die ein Tutorial nötig haben.

Auf mobilen Geräten ist das Tippen auf der Touch-Screen-Tastatur aufwändiger und in der Regel auch langsamer als bei einer mechanischen Tastatur. Als Alternative könnte das

System um eine Sprachaufnahme erweitert werden, die es den Benutzer auf einfache Weise ermöglicht in Interaktion mit den anderen Teilnehmern zu treten.

Da es auf Tablets mehr Platz zur Darstellung von Informationen gibt, wäre die Entwicklung eines eigenen Designs für Tablets eine gute Idee für ein zukünftiges Projekt. Durch das verwenden des *Sencha Touch* Frameworks kann dabei auf das vorhandene System aufgesetzt werden.

Als nützliche Erweiterung könnte sich auch eine automatische Synchronisation im Hintergrund des mobilen Betriebssystems erweisen. Durch die Möglichkeit der Benachrichtigungen auf mobilen Geräten könnte auf den Umweg über E-Mails verzichtet und der Benutzer mittels Klingelton oder Vibration des Gerätes auf bestimmte Ereignisse hingewiesen werden.

Wie im Kapitel 4.3 über den Benutzerbereich angemerkt, macht der Einsatz von anonymen Benutzern auf einem Smartphone das System unnötig kompliziert. Daher wäre ein zukünftiges Projekt, die Entfernung des anonymen Benutzers vom System und die Vereinfachung der Registrierung neuer Benutzer.

Abbildungsverzeichnis

1.1	Global Internet Connected Device Shipments [Worldwide Smart Connected, 2013]	3
1.2	Weltweite Smartphone-Marktprognose 2013-2017 [Worldwide Mobile Phone, 2013]	4
2.1	Hauptseite der WeDecide Web-Version	11
2.2	Beschreibung von Hybrid Apps [Hybrid Apps, 2014]	17
3.1	Architektur des Systems <i>WeDecideMobile</i>	19
3.2	Model View Controller Entwurfsmuster [Simple Example of MVC, 2008] .	21
3.3	Neue Themes seit Sencha Touch 2.3 [Agrawal, 2013]	28
3.4	Beispiel für ein Verhalten in Cucumber [Cucumber Division Feature, 2014]	32
4.1	Hauptseite	34
4.2	Buttons des Hauptmenüs	35
4.3	Detailansicht einer Gruppenentscheidungsaufgabe	36
4.4	Teilnehmer hinzufügen	37
4.5	Link für anonymen Benutzer	37
4.6	Forum einer Gruppenentscheidung	38
4.7	Anmeldung eines Benutzers	39
4.8	Profilverwaltung	39
4.9	Registrierung eines Benutzers	40
4.10	Zurücksetzen des Passworts	40
4.11	Diagramm zum Ablauf der Modellierung einer Gruppenentscheidungsaufgabe	42
4.12	Phasendiagramm des Gruppenentscheidungsprozesses	43
4.13	Basis-Formular einer Modellierung	44
4.14	Picker für die Gruppenentscheidungsart	45

4.15	Spezifische Einstellungen des „Antworten finden“ Szenarios	46
4.16	Fragenliste des „Antworten finden“ Szenarios	47
4.17	Frageneditor des „Antworten finden“ Szenarios	48
4.18	Finalisierung einer Modellierung	49
4.19	Phase 1 des „Antworten finden“ Szenarios	50
4.20	Phase 2 des „Antworten finden“ Szenarios	51
4.21	Abschluss des „Antworten finden“ Szenarios	53
4.22	Spezifische Einstellungen des „Wähle eine Alternative“ Szenarios	55
4.23	Modellierung der Alternativen des „Wähle eine Alternative“ Szenarios	56
4.24	Phase 1 des „Wähle eine Alternative“ Szenarios	57
4.25	Phase 2 des „Wähle eine Alternative“ Szenarios	58
4.26	Abschluss des „Wähle eine Alternative“ Szenarios	59
4.27	Spezifische Einstellungen des „Suche & wähle eine Alternative“ Szenarios	61
4.28	Modellierung der Alternativen des „Suche & wähle eine Alternative“ Szenarios	62
4.29	Phase 1 des „Suche & wähle eine Alternative“ Szenarios	63
4.30	Phase 2 des „Suche & wähle eine Alternative“ Szenarios	64
4.31	Abschluss des „Suche & wähle eine Alternative“ Szenarios	65
4.32	Spezifische Einstellungen des „Einen Termin finden“ Szenarios	67
4.33	Zeiteinteilung des „Einen Termin finden“ Szenarios	68
4.34	Phase 1 des „Einen Termin finden“ Szenarios	69
4.35	Phase 2 des „Einen Termin finden“ Szenarios	70
4.36	Abschluss des „Einen Termin finden“ Szenarios	71
4.37	Spezifische Einstellungen des „Anforderungen priorisieren“ Szenarios	74
4.38	Modellierung der Anforderungen des „Anforderungen priorisieren“ Szenarios	75
4.39	Phase 1 des „Anforderungen priorisieren“ Szenarios	76
4.40	Phase 2 des „Anforderungen priorisieren“ Szenarios	77
4.41	Abschluss des „Anforderungen priorisieren“ Szenarios	78
4.42	Evaluierung mittels System Usability Scale	79

Abkürzungsverzeichnis

ASF	Apache Software Foundation
BDD	Behavior Driven Development
CSS	Cascading Style Sheets
CSS3	Cascading Style Sheets 3
DOM	Document Object Model
GUI	Graphical User Interface
i18n	Internationalisierung
IDC	International Data Corporation
HTML	Hypertext Markup Language
HTML5	Hypertext Markup Language 5
HTTP	Hypertext Transfer Protocol
JS	JavaScript
JSON	JavaScript Object Notation
JSONP	JSON <i>with Padding</i>
JSP	Java Server Pages
MVC	Model View Controller
SASS	Syntactically Awesome Stylesheets
SOP	Same-Origin-Policy

- SUS** System Usability Scale
- TDD** Test Driven Development
- UI** User Interface
- UUID** Universally Unique Identifier
- XML** Extensible Markup Language

Literaturverzeichnis

15 Jahre WWW, 2008. 15 Jahre WWW: Die Browserkriege. (<http://www.golem.de/0805/59377.html>), 2008. Online, Last accessed on 2014-03-12.

About Apache Cordova, 2014. About Apache Cordova. (<https://cordova.apache.org/>), 2014. Online, Last accessed on 2014-03-12.

G. Agrawal. Announcing Sencha Touch 2.3 - Touch Grid, Cordova Support, and New Themes. (<http://www.sencha.com/blog/announcing-sencha-touch-2-3-html5-for-ios7-touch-grid-cordova-support-more/>), 2013. Online, Last accessed on 2014-03-12.

Apache Lucene, 2013. Apache lucene. <http://lucene.apache.org/>, 2013. [Online, Last accessed on 2013-11-17].

Apache Tomcat, 2013. Apache tomcat. <http://tomcat.apache.org/>, 2013. [Online, Last accessed on 2013-11-17].

L. Ardissono, A. Goy, G. Petrone, M. Segnan und P. Torasso. Intrigue: Personalized recommendation of tourist attractions for desktop and hand held devices. *Applied Artificial Intelligence*, 17(8-9):687–714, 2003.

E. Aronson, T. Wilson und R. Akert. *Social Psychology*. Pearson Prentice Hall, 2005.

A. Bangor, P. T. Kortum und J. T. Miller. An empirical evaluation of the system usability scale. *International Journal of Human-Computer Interaction*, 24(6):574–594, 2008.

T. Bhat und N. Nagappan. Evaluating the efficacy of test-driven development: Industrial case

- studies. In *Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering*, pages 356–363, New York, NY, USA, 2006. ACM.
- R. D. Burke, K. J. Hammond und B. C. Young. Knowledge-based navigation of complex information spaces. In W. J. Clancey und D. S. Weld, editors, *AAAI/IAAI, Vol. 1*, pages 462–468. AAAI Press / The MIT Press, 1996.
- D. L. Chao, J. Balthrop und S. Forrest. Adaptive radio: Achieving consensus using negative preferences. In *Proceedings of the 2005 International ACM SIGGROUP Conference on Supporting Group Work, Sanibel Island, Florida, United States*, pages 120–123. ACM SIGGROUP conference on Supporting group work, ACM, 2005.
- Y. H. Cho, J. K. Kim und S. H. Kim. A personalized recommender system based on web usage mining and decision tree induction. *Expert Syst. Appl.*, 23(3):329–342, 2002.
- I. A. Christensen und S. N. Schiaffino. Entertainment recommender systems for group of users. *Expert Syst. Appl.*, 38(11):14127–14135, 2011.
- A. Crossen, J. Budzik und K. J. Hammond. Flytrap: intelligent group music recommendation. In *IUI*, pages 184–185, 2002.
- Cucumber, 2014. Cucumber. (<http://cukes.info/>), 2014. Online, Last accessed on 2014-03-12.
- Cucumber Division Feature, 2014. Cucumber Division Feature. (<https://github.com/cucumber/cucumber/blob/master/examples/i18n/de/features/division.feature>), 2014. Online, Last accessed on 2014-03-12.
- DB-Engines Ranking, 2014. DB-Engines Ranking. (<http://db-engines.com/en/ranking>), 2014. Online, Last accessed on 2014-03-12.
- A. Felfernig und R. Burke. Constraint-based recommender systems: technologies and research issues. In *Proceedings of the 10th International Conference on Electronic Commerce, ICEC '08*, pages 1–10, New York, NY, USA, 2008. ACM.
- A. Felfernig, C. Zehentner, G. Ninaus, H. Grabner, W. Maalej, D. Pagano, L. Weninger und

- F. Reinfrank. Group decision support for requirements negotiation. In L. Ardissono und T. Kuflik, editors, *UMAP Workshops*, volume 7138 of *Lecture Notes in Computer Science*, pages 105–116. Springer, 2011.
- E. Freeman und E. Freeman. *Entwurfsmuster von Kopf bis Fuß*. O'Reilly, Köln, 2006.
- S. V. Gagern. Native vs. Web App vs. Hybrid: Was ist die perfekte Entwicklerstrategie? (<https://www.developergarden.com/de/blog/artikel/article/native-vs-web-app-vs-hybrid-was-ist-die-perfekte-entwicklerstrategie/>), 2013. Online, Last accessed on 2014-03-12.
- I. Garcia, S. Pajares, L. Sebastia und E. Onaindia. Preference elicitation techniques for group recommender systems. *Inf. Sci.*, 189:155–175, 2012.
- Gavin King, 2004. The Interview: Gavin King, Hibernate. (<http://www.javaperformancetuning.com/news/interview041.shtml>), 2004. Online, Last accessed on 2014-03-12.
- D. Goldberg, D. Nichols, B. M. Oki und D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.
- D. Goren-Bar und O. Glinansky. Family stereotyping - a model to filter tv programs for multiple viewers. In *Proceedings of the Workshop on Personalization in Future TV at the Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH-2002)*, pages 101–108, Malaga, Spain, 2002. Springer Verlag.
- H. Heitkötter, S. Hanschke und T. A. Majchrzak. Comparing cross-platform development approaches for mobile applications. In K.-H. Krempels und J. Cordeiro, editors, *WEBIST*, pages 299–311. SciTePress, 2012.
- S. Herr, A. Rösch, C. Beckmann und T. Gross. Informing the design of group recommender systems. In J. A. Konstan, E. H. Chi und K. Höök, editors, *CHI Extended Abstracts*, pages 2507–2512. ACM, 2012.
- Hibernate, 2013. Hibernate. <http://hibernate.org/>, 2013. [Online, Last accessed

on 2013-11-17].

Hibernate ORM, 2012. Hibernate ORM 4.1.9.Final Released. (<http://in.relation.to/23792.lace>), 2012. Online, Last accessed on 2014-03-12.

Hybrid Apps, 2014. Hybrid Apps. (http://commons.wikimedia.org/wiki/File:Hybrid_Apps.jpg), 2014. Online, Last accessed on 2014-03-12.

Introducing JSON, 2014. Introducing JSON. (<http://www.json.org/>), 2014. Online, Last accessed on 2014-03-12.

A. Jameson. More than the sum of its members: challenges for group recommender systems. In M. F. Costabile, editor, *AVI*, pages 48–54. ACM Press, 2004.

A. Jameson und B. Smyth. Recommendation to groups. In P. Brusilovsky, A. Kobsa und W. Nejdl, editors, *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*, pages 596–627. Springer, 2007.

Jasmine, 2013. Jasmine. (<http://jasmine.github.io/2.0/introduction.html>), 2013. Online, Last accessed on 2014-03-12.

JavaScript Object Notation, 2012. JavaScript Object Notation with Padding. (<http://jsonp.eu/>), 2012. Online, Last accessed on 2014-03-12.

S. Koch. *JavaScript - Einführung, Programmierung und Referenz*. Dpunkt Verlag, 6 edition, 2011.

A. Kosmaczewski. *Mobile JavaScript Application Development - Bringing Web Programming to Mobile Devices*. O'Reilly, 2012.

A. Kosmaczewski. *Sencha Touch 2 Up and Running*. O'Reilly Media, 2013.

G. E. Krasner und S. T. Pope. A cookbook for using the model-view controller user interface paradigm in smalltalk-80. *Journal of Object-Oriented Programming*, 1(3):26 – 49, Aug. 1988.

- S. K. Lee, Y. H. Cho und S. H. Kim. Collaborative filtering with ordinal scale-based implicit ratings for mobile music recommendations. *Inf. Sci.*, 180:2142–2155, June 2010.
- A. Leff und J. T. Rayfield. Web-application development using the model/view/controller design pattern. In *IEEE Enterprise Distributed Object Computing Conference*, pages 118–127, Seattle, Washington USA, September 2001.
- H. Lieberman, N. W. V. Dyke und A. S. Vivacqua. Let's browse: A collaborative web browsing agent. In *IUI*, pages 65–68, 1999.
- R. Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 22(140): 1–55, 1932.
- T. M. H. Reenskaug. The model-view-controller (mvc) – its past and present. http://home.ifi.uio.no/trygver/2003/javazone-jaoo/MVC_pattern.pdf, Aug. 2003. URL http://home.ifi.uio.no/trygver/2003/javazone-jaoo/MVC_pattern.pdf.
- R. C. Martin und J. O. Coplien. *Clean code: a handbook of agile software craftsmanship*. Prentice Hall, Upper Saddle River, NJ [etc.], 2009.
- J. Masthoff. Group modeling: Selecting a sequence of television items to suit a group of viewers. *User Model. User-Adapt. Interact.*, 14(1):37–85, 2004.
- J. Masthoff. Group recommender systems: Combining individual models. In F. Ricci, L. Rokach, B. Shapira und P. B. Kantor, editors, *Recommender Systems Handbook*, pages 677–702. Springer, 2011.
- J. F. McCarthy. Pocket restaurant finder: A situated recommender systems for groups. In *Proceeding of Workshop on Mobile Ad-Hoc Communication at the 2002 ACM Conference on Human Factors in Computer Systems*, 2002.
- J. F. McCarthy und T. D. Anagnost. Musicfx: An arbiter of group preferences for computer supported collaborative workouts. In S. E. Poltrock und J. Grudin, editors, *CSCW*, pages 363–372. ACM, 1998.

- K. McCarthy, M. Salamó, L. Coyle, L. McGinty, B. Smyth und P. Nixon. Cats: A synchronous approach to collaborative group recommendation. In G. Sutcliffe und R. Goebel, editors, *FLAIRS Conference*, pages 86–91. AAAI Press, 2006.
- B. N. Miller, I. Albert, S. K. Lam, J. A. Konstan und J. Riedl. Movielens unplugged: experiences with an occasionally connected recommender system. In *IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces*, pages 263–266, 2003.
- S. Min und I. Han. Detection of the customer time-variant pattern for improving recommender systems. *Expert Syst. Appl.*, 28(2):189–199, 2005.
- MySQL, 2013. Mysql. <http://www.mysql.com/>, 2013. [Online, Last accessed on 2013-11-17].
- D. North. Introducing BDD. *Better Software Magazine*, Mar. 2006.
- M. O'Connor, D. Cosley, J. A. Konstan und J. Riedl. PolyLens: A recommender system for groups of user. In W. Prinz, M. Jarke, Y. Rogers, K. Schmidt und V. Wulf, editors, *ECSCW*, pages 199–218. Kluwer, 2001.
- M. J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artif. Intell. Rev.*, 13(5-6):393–408, 1999.
- T. Pessemier, S. Dooms und L. Martens. Comparison of group recommendation algorithms. *Multimedia Tools and Applications*, pages 1–45, 2013.
- PhoneGap, Cordova, 2012. PhoneGap, Cordova, and what's in a name? (<http://phonegap.com/2012/03/19/phonegap-cordova-and-what%E2%80%99s-in-a-name/>), 2012. Online, Last accessed on 2014-03-12.
- P. Resnick, N. Iacovou, M. Suchack, P. Bergstrom und J. Riedl. GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, pages 175–186, 1994.
- Same-origin policy, 2013. Same-origin policy. <https://developer.mozilla.org/>

- en-US/docs/Web/JavaScript/Same_origin_policy_for_JavaScript, 2013. [Online, Last accessed on 2014-01-06].
- J. B. Schafer, D. Frankowski, J. L. Herlocker und S. Sen. Collaborative filtering recommender systems. In P. Brusilovsky, A. Kobsa und W. Nejdl, editors, *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*, pages 291–324. Springer, 2007.
- Sencha Architect Guide, 2014. Architect 2 Sencha Docs. (<http://docs.sencha.com/architect/2/>), 2014. Online, Last accessed on 2014-03-12.
- Sencha Inc. Sencha Touch Build Mobile Web Apps with HTML5 Version 2.3. <http://www.sencha.com/products/touch/>, 2013. [Online, Last accessed on 2013-11-17].
- Sencha Touch Licensing, 2014. Sencha Touch Licensing. (<http://www.sencha.com/products/touch/license/>), 2014. Online, Last accessed on 2014-03-12.
- Servlet, 2014. Servlet. (<http://www.itwissen.info/definition/lexikon/Servlet.html>), 2014. Online, Last accessed on 2014-03-12.
- Simple Example of MVC, 2008. Simple Example of MVC (Model View Controller) Design Pattern for Abstraction. (<http://www.codeproject.com/Articles/25057/Simple-Example-of-MVC-Model-View-Controller-Design>), 2008. Online, Last accessed on 2014-03-12.
- M. Stettinger. *Intelligent Services for Group Decision Making*. Diplomarbeit, Graz, 2013.
- M. Stettinger, G. Ninaus, M. Jeran, F. Reinfrank und S. Reiterer. We-decide: A decision support environment for groups of users. In M. Ali, T. Bosse, K. V. Hindriks, M. Hoogendoorn, C. M. Jonker und J. Treur, editors, *IEA/AIE*, volume 7906 of *Lecture Notes in Computer Science*, pages 382–391. Springer, 2013.
- The Tomcat Story, 2014. The Tomcat Story. (<http://tomcat.apache.org/heritage.html>), 2014. Online, Last accessed on 2014-03-12.
- B. Towle und C. Quinn. Knowledge based recommender systems using explicit user models.

- In *Papers from the AAAI Workshop, AAAI Technical Report WS-00-04*, pages 74–77. Menlo Park, CA: AAAI Press, 2000.
- T. Tullis und B. Albert. *Measuring the User Experience: Collecting, Analyzing, and Presenting Usability Metrics*). Morgan Kaufmann, 2008.
- C. Ullenboom. Java ist auch eine Insel. http://openbook.galileocomputing.de/javainsel9/javainsel_23_012.htm, 2011.
- M. G. Vozalis und K. G. Margaritis. Using svd and demographic data for the enhancement of generalized collaborative filtering. *Inf. Sci.*, 177(15):3017–3037, 2007.
- C. Wenz. *JavaScript und AJAX*. Galileo Computing, Bonn, 7., aktualisierte auflage edition, 2007.
- Worldwide Mobile Phone, 2013. Worldwide Mobile Phone Market Forecast to Grow 7.3% in 2013 Driven by 1 Billion Smartphone Shipments, According to IDC . (<http://www.idc.com/getdoc.jsp?containerId=prUS24302813>), 2013. Online, Last accessed on 2014-03-12.
- Worldwide Smart Connected, 2013. Worldwide Smart Connected Device Market Crossed 1 Billion Shipments in 2012, Apple Pulls Near Samsung in Fourth Quarter, According to IDC. <https://www.idc.com/getdoc.jsp?containerId=prUS24037713>, 2013. Online, Last accessed on 2014-03-12.
- Z. Yu, X. Zhou, Y. Hao und J. Gu. Tv program recommendation for multiple viewers based on user profile merging. *User Model. User-Adapt. Interact.*, 16(1):63–82, 2006.