HOCHSCHULE DER MEDIEN

Usability prototypr

Projektdokumentation

des Software-Projektes im WS07/08

Vorgelegt von:	Stefan Nolte
E-Mail Adresse:	sn015@hdm-stuttgart.de
Matrikel-Nr.:	15710
Eingereicht am:	29.02.2008
Betreuer:	Prof. Dr. Johannes Maucher
Studiengang:	Medieninformatik

Hochschule der Medien
Nobelstraße 10
70569 Stuttgart-Vaihingen

Inhaltsverzeichnis

Abbildungsverzeichnis

Codebeispiele

Abkürzungsverzeichnis

bzw.	beziehungsweise
ca.	circa
Dr.	Doktor
GPS	Global Positioning System
IDE	Integrated Development Enviroment
LBS	Location Based Services
LBSN	Location Based Social Network
POI	Point of Interest
Prof.	Professor

1 Einleitung und Aufgabenbeschreibung

Dieses Projekt wurde begleitend zur Bachelor-Thesis durchgeführt und umfasst die Entwicklung einer Benutzungsschnittstelle für eine mobile kartenbasierte Applikation.

1.1 Aufgabenumfeld

Die Anwendung, für die die Benutzungsschnittstelle entworfen wurde, ist eine mobile Applikation eines Location Based Social Networks (im Folgenden LBSN) mit dem Namen *flamr* (siehe Abschnitt 3). Ein solches LBSN verbindet die Elemente der Social Communities mit den Möglichkeiten der Location Based Services (im Folgenden LBS, siehe Kapitel 2). Diese Applikation sollte auf aktuellen Mobiltelefonen laufen und dort komfortabel bedient werden können.

Um eine gebrauchstaugliche Bedienung zu ermöglichen, sind im Rahmen der Bachelor-Thesis Gestaltungsvorschläge entstanden, die in diesem Projekt in einen Prototyp umgesetzt und so visualisiert werden sollen. Dabei steht dieser Prototyp in keiner Verbindung zur letztendlichen Implementierung, sondern sollte nur die grobe Informationsdarstellung und die ungefähren Interaktionsabläufe enthalten. Auf dieser Basis kann der Prototyp evaluiert und die verschiedenen Gestaltungsvorschläge auf ihre Gebrauchstauglichkeit untersucht werden. Eine komplexe Implementierung würde den Zeitaufwand nicht rechtfertigen, da die Wahrscheinlichkeit hoch ist, dass der Prototyp im Laufe der Gestaltung geändert oder verworfen wird.

1.2 Über diese Dokumentation

Diese Dokumentation enthält zum Teil direkt übernommene Passagen aus der begleitenden Bachelor-Thesis [1]. Diese wurden in dieses Dokument integriert, damit der Leser einen schnellen Überblick über den Rahmen dieser Projektarbeit bekommt und dazu nicht die Bachelor-Thesis lesen muss.

Um den Rahmen dieser Arbeit nicht zu sprengen, sollen die Gestaltungsvorschläge an dieser Stelle nicht beschrieben werden. Stattdessen wird auf die Bachelor-Thesis [1] verwiesen.

1.3 Aufbau dieser Dokumentation

Im ersten Teil dieser Dokumentation soll das Umfeld dieses Projektes erläutert werden. Dazu wird zunächst der technologische Rahmen beschrieben und die Applikation flamr vorgestellt, für die das User Interface konzipiert wurde. Im Anschluss daran wird ein kurzer Einblick in die Usability und den benutzerzentrierten Gestaltungsprozess gegeben, damit die Implementierung des Prototyps in den Gesamtkontext der Entwicklung eingeordnet werden kann.

Der zweite Teil beinhaltet die Erläuterung der Umsetzung der Gestaltungsvorschläge. Dazu werden die verschiedenen Möglichkeiten, die zur Auswahl standen kurz evaluiert und die Entwicklungsumgebung vorgestellt. Anschließend wird der Flash-Prototyp beschrieben und auf dessen Implementierung und die Softwarearchitektur eingegangen.

2 Die technologische Umgebung von flamr

In diesem Kapitel sollen die technologischen Begrifflichkeiten erläutert werden, um den Kontext der Applikation flamr zu verdeutlichen. Dazu werden zunächst die Social Communities und die Möglichkeiten der Navigation auf mobilen Endgeräten beschrieben. Im Anschluss daran werden die Location Based Services und die Thematik der Location Based Social Networks beleuchtet.

2.1 Social Communities

Eine Social Community ist ein Verbund aus mehreren Nutzern, die sich über eine Plattform austauschen. Diese Plattform bildet meist eine interaktive Website, jedoch kann dies auch ein Forum sein. Die Nutzer haben zumeist ähnliche Absichten, die sie mit der Mitgliedschaft in der Community verfolgen, beispielsweise wollen diese ihr Wissen in einer Programmiersprache erweitern. Communities können jedoch auch andere Themen beinhalten, zum Beispiel bewerten die Nutzer gegenseitig ihre Fotos oder tauschen sich über bestimmte Veranstaltungen aus.

Die Basis für eine solche Community bildet ein Forum. Dieses wird zumeist um viele Funktionen erweitert, beispielsweise haben die Nutzer eigene Profile, auf denen sie sich vorstellen können oder es existiert ein Nachrichtensystem, um sich gegenseitig Mails schicken zu können.

2.2 Navigationsmöglichkeiten auf Mobiltelefonen

In immer mehr Mobiltelefonen, die auf dem Markt erscheinen, ist ein GPS-Modul[1] integriert. Die Genauigkeit der Ortung liegt hier bei ca. 15 Metern. Eine weitere Möglichkeit zur Lokalisierung besteht darin, dass der Standort des Nutzers durch die Funkzelle ermittelt wird, von der dessen Mobiltelefon angesprochen wird. Da diese Variante somit von der Größe der Funkzelle abhängt, ist die Messung relativ ungenau. In Großstädten kann zwar eine Genauigkeit von wenigen hundert Metern erwartet werden, jedoch können vor allem in der ländlichen Region Messfehler von mehreren Kilometern auftreten, da die Funkzellen dort zumeist relativ groß sind.

[1] GPS ist ein satellitengestütztes System zur Positionsbestimmung.

Durch die Kombination mit einem Kartensystem kann das Mobiltelefon so als Navigationssystem genutzt werden. Dass dies ein großer Trend ist, zeigt die Übernahme des Anbieters von Geodaten *Navteq*: Für dessen Akquisition war der Telekommunikationskonzern *Nokia* bereit, rund acht Milliarden Dollar zu bezahlen (teilweise Auszug aus der Bachelor-Thesis [1]).

2.3 Location Based Services

Location Based Services oder Standortbezogene Dienste sind mobile Dienste, welche dem Nutzer bestimmte Informationen abhängig vom Kontext, in dem dieser sich befindet, bereitstellen. Der Kontext ist wiederum von der aktuellen Zeit, der Position und der Persönlichkeit des Nutzers abhängig. Zusätzlich dazu können die Informationen mit anderen Diensten verknüpft werden, so dass neue Inhalte und somit ein Mehrwert entsteht. Diese Verknüpfung wird allgemein als Mashup bezeichnet.

Ein aktuelles Beispiel für LBS ist die mobile Arbeitszeiterfassung, bei der die Arbeitszeiten der Außendienstmitarbeiter durch deren Position erfasst werden. Eine andere Form von LBS ist ein Stadtführer, der interessante Orte in der Nähe aufzeigt, die wiederum von Benutzern eingepflegt werden können. Diese Orte werden Points of Interest (folgend POI) genannt und umfassen Restaurants, Hotels, Geschäfte, Apotheken, Bankautomaten, Tankstellen, Sehenswürdigkeiten und viele weitere Punkte, die für den Nutzer von Interesse sein könnten (Auszug aus der Bachelor-Thesis [1]).

2.4 Location Based Social Network

Ein Location Based Social Network kombiniert den Aspekt der Social Communities mit den Location Based Services. Dieser Begriff befindet sich momentan in der Entstehungsphase und hat sich daher im allgemeinen Sprachgebrauch noch nicht durchgesetzt.

Ein LBSN besteht aus zwei Komponenten. Die Basis ist die Social Community im Web, welche sich inhaltlich mit einem Mashup aus Kartendaten mit anderen Inhalten befasst. Zusätzlich zu dem Zugang über das Internet kann mobil auf die Funktionen der Plattform zugegriffen werden, wobei die LBS eine signifikante Rolle einnehmen (Auszug aus der Bachelor-Thesis [1]).

Ein Beispiel für ein solche LBSN ist die Anwendung flamr, die im nächsten Kapitel beschrieben wird.

3 Die Applikation flamr

In diesem Kapitel soll nun die Anwendung flamr vorgestellt werden, für dessen mobile Applikation die Benutzungsschnittstelle entworfen wurde.

Flamr ist ein spezielles Location Based Social Network für Studenten und Auszubildende. Die Grundidee dabei ist, dass der Nutzer durch flamr in der Lage sein soll, seine Freizeitaktivitäten komfortabel und flexibel zu planen. So sollen durch diese Plattform alle Aspekte, die für die Freizeitplanung nötig sind, abgedeckt werden.

Die Basis bildet auch hier die Online-Community, in der ein Nutzer sein eigenes Profil besitzt und mit anderen Nutzern kommunizieren kann. Zusätzlich dazu können Treffen mit anderen Nutzern arrangiert oder Mitfahrgelegenheiten organisiert werden. Weiterhin besteht die Möglichkeit, sich über verschiedene Points of Interest und Veranstaltungen zu informieren, diese zu editieren oder zu bewerten (teilweise Auszug aus der Bachelor-Thesis [1]).

Diese Funktionen umfassen zumeist eine Kartendarstellung: Die verschiedenen POIs und Veranstaltungen können auf einer Karte betrachtet und die Kontakte des Nutzers auf dieser lokalisiert werden. In Analogie zur mobilen Navigation (siehe Kapitel 2.2) sollte der Nutzer auch Routen zu bestimmten Punkten berechnen können.

Diese Funktionalität soll sowohl über die Webapplikation als auch die mobile Applikation zugänglich sein. Der Vorteil in der mobilen Variante liegt darin, dass der Nutzer somit in seiner Planung ungebunden und flexibel ist und dadurch spontaner agieren kann.

4 Usability

Dieses Kapitel beinhaltet einen kleinen Einblick in die Usability und den Gestaltungsprozess, mit dem eine Benutzungsschnittstelle gebrauchstauglich gestaltet werden kann.

4.1 Was ist Usability?

Usability ist die Gebrauchstauglichkeit eines Produktes und bezeichnet dessen Eignung, wie effektiv, effizient und zufrieden stellend ein bestimmter Nutzer mit diesem in einem bestimmten Nutzungskontext seine Ziele erreichen kann.

4.2 Der benutzerzentrierte Gestaltungsprozess

Der benutzerzentrierte Gestaltungsprozess ist ein iterativer Entwicklungsprozess, mit dem eine Applikation gebrauchstauglich gestaltet werden kann. Die Besonderheit an dieser Vorgehensweise besteht darin, dass der Nutzer stets Maßstab für alle Gestaltungsentscheidungen ist und aktiv an der Gestaltung des User Interfaces teilnimmt.

Im Einzelnen besteht der Gestaltungsprozess aus vier zentralen Phasen:

1. In der **Analyse des Nutzungskontextes** wird der Benutzer mit seinen Aufgaben und der Umgebung, in der das Produkt eingesetzt wird, analysiert. Aus dieser Analyse entstehen Anforderungen an die Benutzungsschnittstelle, die in der weiteren Gestaltung berücksichtigt werden sollten.
2. Die **Entwurfs- und Gestaltungsphase** dient dazu, auf Basis der Informationen und Anforderungen der Nutzungskontextanalyse ein durchgängig konsistentes Designkonzept zu entwerfen. Dabei gliedert sich diese Phase selber in drei Abschnitte: Im ersten Schritt wird im Funktionsdesign die Funktionalität festgelegt, ohne auf die Eigenschaften der Benutzungsschnittstelle einzugehen. In der zweiten Phase des Informationsdesigns werden die Elemente der Präsentationsschicht entwickelt. Im dritten Abschnitt sollen die Interaktionen festgelegt werden, die bei der Bedienung der Applikation auftreten.
3. Die dritte Phase beschäftigt sich mit dem **Prototyping**. Hier werden die Gestaltungsentscheidungen, die in der zweiten Phase getroffen wurden, in ein leicht erlebbares und kommunizierbares Modell umgesetzt. Es sollte so wenig

Arbeit wie möglich in den Prototyp gesteckt werden, um diesen bei Bedarf schnell verwerfen zu können. Weiterhin sollte ein Prototyp nur die Bereiche beinhalten, die wirklich in der vierten Phase, der Evaluation benötigt werden.

4. In der **Evaluation** wird das Design anhand des Prototyps beurteilt und notwendige Optimierungspotenziale zusammengetragen. Die Basis hierfür bilden die Anforderungen aus der Analyse des Nutzungskontextes. Das Ergebnis dieser Phase ist ein Evaluationsbericht, in dem die Resultate der Evaluation festgehalten sind.

Nach der Evaluationsphase wird auf Basis des Evaluationsberichtes eine Entscheidung getroffen, ob die vier Phasen des Gestaltungsprozesses nochmals durchlaufen werden sollen. Falls sich für einen nächsten Durchgang entschieden wird, müssen die jeweiligen Phasen auf die Ergebnisse angepasst werden (teilweise Auszug aus der Bachelor-Thesis [1]).

5 Möglichkeiten zur Umsetzung des Prototyps

Der benutzerzentrierte Gestaltungsprozess, der im vorigen Kapitel vorgestellt wurde, fand im Rahmen der Bachelor-Thesis und des Projektes Anwendung. In der Thesis wurden die ersten beiden Phasen dieses Prozesses durchgeführt und so der Nutzungskontext analysiert und die Benutzungsschnittstelle entworfen. Die Ideen des Informations- und Interaktionsdesigns (siehe Kapitel 4.2) sollten nun im Rahmen des Projektes in einen Prototyp umgesetzt werden. Hierbei bestehen verschiedene Möglichkeiten zur Umsetzung: Die Gestaltungsvorschläge könnten in einem Papier-Prototyp, mit Powerpoint oder in Flash realisiert werden. Diese verschiedenen Möglichkeiten mussten nun evaluiert werden, um eine angemessene Form der Umsetzung zu finden.

Der Papierprototyp bietet eine schnelle Möglichkeit, bestimmte Aspekte der Benutzungsschnittstelle zu visualisieren und so zu testen. Die Interaktionsabläufe in einem solchen Prototyp müssen jedoch simuliert werden. Dies bedingt, dass der Moderator Kenntnisse über diese Abläufe besitzt. Da die Evaluation der Benutzungsschnittstelle nicht Teil dieses Projektes ist, sollte jedoch ein Prototyp existieren, der keine Fachkenntnis des Moderators erfordert. Aus diesem Grund ist in diesem Kontext ein Papierprototyp nicht geeignet.

Mit Powerpoint ist es möglich, Interaktionsabläufe darzustellen, ohne diese simulieren zu müssen, aber diese Form eines Prototyps ist immer noch relativ statisch und die Möglichkeiten zur Bedienung sind begrenzt.

Ein Flash-Prototyp erfordert zwar mehr Fachkenntnis, jedoch können durch diesen die Gestaltungsvorschläge relativ realitätsnah getestet werden. Durch die WYSIWYG-Entwicklungsumgebung[2] von Flash können schnell Ergebnisse erzielt werden, die durch die Programmierung mit ActionScript 3.0 erweiterbar sind. Zudem existierten in diesem Projekt schon Kenntnisse der objektorientierten Programmierung in Java, so dass eine Einarbeitung in die ebenfalls objektorientierte Programmiersprache ActionScript 3.0 keine lange Zeit erfordert. Aus diesem Grund wurde als Plattform zur Umsetzung der Gestaltungsideen Flash mitsamt ActionScript 3.0 gewählt.

[2] WYSIWYG steht für „**W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et“ und bezeichnet einen Editor, in dem während der Bearbeitung das Dokument so angezeigt wird, wie es bei der Ausgabe aussehen wird.

6 Die Entwicklungsumgebung

In diesem Kapitel soll die Entwicklungsumgebung beschrieben werden. Dabei wird zunächst der organisatorische Rahmen erläutert und anschließend auf die Gestaltungs- und Programmierumgebung eingegangen.

6.1 Die Organisatorische Umgebung

Zur Verwaltung dieses Projektes wurde das Projektmanagementtool *trac* inklusive mehrerer Plugins angewandt. In diesem Tool können Aufgaben als Tickets erstellt, priorisiert und den Projektteilnehmern zugewiesen werden. Weiterhin können Milestones geplant werden. Diese Applikation ist mit einem *SVN Repository*[3] verknüpft. Somit besteht die Möglichkeit, durch Kommentare beim Commit in das Repository zugleich die Tickets zu schließen. Für die Dokumentation von Problemen, Literaturauszügen und Meeting-Ergebnissen wurde ein Wiki[4] integriert, in das diese Informationen eingetragen werden konnten.

Zur Absprache innerhalb des Projektteams wurde ein wöchentlicher Termin vereinbart, in dem alle Teammitglieder ihre Ergebnisse und Probleme vorstellten. Weiterhin wurden in diesem sog. *Jour fixe* allgemein relevante Themen besprochen.

6.2 Die Gestaltungs- und Programmierumgebung

Als Entwicklungsumgebung für die Realisierung der Gestaltungsvorschläge wurde *Adobe Photoshop CS3*, *Adobe Flash CS3* und *Flashdevelop* genutzt.

In Adobe Photoshop CS3 wurden die einzelnen Grafiken erstellt. Diese wurden in Adobe Flash CS3 importiert und zu dem Prototyp zusammengesetzt. Den interaktiven Grafiken wurden Klassen zugewiesen, damit diese im Code referenziert werden konnten (siehe Abschnitt 8.1).

---

[3] Subversion, kurz SVN, ist eine Software zur Versionverwaltung von Dateien und Verzeichnissen. Das Repository bezeichnet das zentrale Archiv des SVN Projektes, in dem alle Elemente enthalten sind.

[4] Ein Wiki ist eine webbasierte Applikation, die ein einfaches Erstellen und Editieren von Webseiten ermöglicht und wird aus diesem Grund von verschiedenen Autoren genutzt, um gemeinsam Inhalte zu erarbeiten.

Die Programmierung erfolgte in der OpenSource Entwicklungsumgebung (im Folgenden IDE – *Integrated Development Enviroment*) Flashdevelop. Flashdevelop ist eine der wenigen ActionScript Editoren, mit denen komplexere Projekte mit mehreren Klassen verwaltet werden können. Jedoch bietet diese Applikation wenig Funktionalität zur professionellen Entwicklung von Anwendungen mit ActionScript – beispielsweise existiert kein Debugger, es gibt keine Möglichkeit zur Refaktorisierung des Programmcodes und Programmierfehler werden erst zur Laufzeit angezeigt. Aktuell existieren jedoch wenig kostenlose Alternativen, da ActionScript erst ab der Version 3.0, die im Juni 2006 veröffentlicht wurde, vollständig objektorientiert ist. Das Plugin FDT für die IDE Eclipse bietet zwar weitaus mehr Möglichkeiten, ist aber kostenpflichtig.

7 Der Prototyp

Als Grundlage für die Erläuterung der Implementierung soll in diesem Kapitel der Flash Prototyp vorgestellt werden. Dieser basiert auf einem Screenshot des Mobiltelefon-Emulators des *Sun Java Wireless Toolkits*. Die Bedienung erfolgt über das Steuerkreuz und die drei Softkeys.

Der Prototyp ist in verschiedene Anzeigemodi aufgeteilt: Der Navigation, der Kartendarstellung und der Inhaltsdarstellung. Auf diese soll nun im Folgenden eingegangen werden.

7.1 Die Navigation

Die Navigation erfolgt in einem dynamischen hierarchischen Karussell, welches aus zwei Ebenen besteht. Um zwischen den verschiedenen Menüpunkten hin- und herzuschalten, muss der linke bzw. rechte Pfeil des Steuerkreuzes auf der Mobiltelefonoberfläche betätigt werden. Infolgedessen werden die verschiedenen Menüpunkte automatisch zentriert und durch eine Fish-Eye[5] Optik dynamisch vergrößert. Falls bei einem Menüpunkt, welches ein Submenü am oberen Rand der Menüebene enthält, der mittlere Softkey oder der obere Pfeil des Steuerkreuzes betätigt wurde, wird dieses Submenü angezeigt. Das aktuelle Menü wird dabei dynamisch vergrößert, während die andere Menüebene verkleinert wird. Um sich zu einem Menü dessen Elternmenü anzeigen zu lassen, ist der untere Pfeil des Steuerkreuzes zu drücken. Falls bei einem Menüpunkt kein Submenü angezeigt wird, kann mit Betätigung des mittleren Softkeys der dazugehörige Inhalt geöffnet werden. Dieses soll in der folgenden Abbildung anhand von Screenshots des Prototyps verdeutlicht werden:

[5] Der Fish-Eye View bezeichnet eine Technik, mit der das aktuell fokussierte Element und nahe gelegene Elemente größer und detaillierter dargestellt werden als entfernte Elemente.

Abbildung 1: Die Navigation des Prototyps

7.2 Die Kartendarstellung

Über den Menüpunkt *Map* wird dessen Untermenü erreicht, worin jedes Element auf die Darstellung der Karte verlinkt. Bevor die Karte angezeigt wird, kann der Nutzer mittels Checkboxen auswählen, welche Kategorien von POIs, Events und Nutzern auf der Karte angezeigt werden sollen (siehe Abbildung 2). Dieser Auswahl liegt jedoch noch keine Funktionalität zugrunde und so werden stets dieselben POIs angezeigt.

Abbildung 2: Die Kartenauswahl des Prototyps

In der Kartendarstellung sind fünf POIs visualisiert und am unteren linken Displayrand in einer Legende zusammengefasst. Wenn mittels des Steuerkreuzes die Karte gescrollt wird und das Fadenkreuz über einem POI liegt, erscheint eine Sprechblase zu diesem, die Text über den POI enthält. Durch Betätigen des mittleren Softkeys erreicht der Nutzer nun die Seite des POIs mit verschiedenen Informationen. Im Prototyp verlinkt dies jedoch nur auf die normale Inhaltsdarstellung (siehe Kapitel 7.3). Falls das Fadenkreuz nicht über einem POI liegt, kann durch Betätigen des mittleren Softkeys die Zoom-Funktion aufgerufen werden, in der mit dem oberen und unteren Pfeil auf dem Steuerkreuz herein- bzw. herausgezoomt werden kann. Auch die Zoom-Funktion ist nur beispielhaft integriert und umfasst noch keine Funktionalität, sondern soll nur die Art und Weise skizzieren.

Abbildung 3: Die Kartendarstellung des Prototyps (Sprechblase & Zoomfunktion)

7.3 Die Inhaltsdarstellung

Der Inhalt von z.B. einer POI-Seite oder eines Nutzerprofils ist in verschiedene Register unterteilt. Diese Unterteilung wird im Prototyp visualisiert, ohne auf den genauen Inhalt einzugehen. Durch Betätigen des linken oder rechten Pfeils des Steuerkreuzes kann zwischen den jeweiligen Registern hin- und hergeschaltet werden. Durch Drücken auf den oberen oder unteren Pfeil des Steuerkreuzes können die verschiedenen Links innerhalb einer Registerseite ausgewählt werden. Weitergehend könnte nun durch Betätigen des mittleren Softkeys der Link aufgerufen werden. Diese Funktion ist jedoch im Prototyp noch nicht enthalten. Die verschiedenen Links sind durch Textfelder visualisiert und werden durch Änderung der Hintergrundfarbe hervorgehoben. Zwischen den Links könnte in der letztendlichen Implementierung der Applikation zusätzlich anderer Inhalt auf einer Seite dargestellt werden.

Abbildung 4: Die Inhaltsdarstellung des Prototyps

Nachdem in diesem Kapitel der Flash-Prototyp beschrieben wurde, soll im nächsten Abschnitt auf dessen Implementierung eingegangen werden.

8 Die Implementierung des Prototyps

Dieser Abschnitt befasst sich mit der Implementierung des Prototyps. Dazu wird zunächst die allgemeine Interaktion zwischen Flash CS3 und Actionscript 3.0 beschrieben. Im Anschluss daran soll die Softwarearchitektur des Prototyps anhand eines vereinfachten UML-Diagramms erläutert werden.

8.1 Zusammenspiel von Flash CS3 und Actionscript 3.0

In der Entwicklungsumgebung von Flash CS3 wurde die grundlegende grafische Gestaltung vollzogen und mit dem ActionScript Quelltext verknüpft. So wurde in einem ersten Schritt die Oberfläche eines Mobiltelefons gestaltet, auf der der Prototyp bedient werden soll.

Damit die verschiedenen Tasten der Mobiltelefonoberfläche betätigt werden können, wurde unter die Ebene des Mobiltelefon-Screenshots eine weitere Ebene eingefügt, auf der sich Schaltflächen befinden. Diese Schaltflächen sind Instanzen eines Symbols der Flash-Bibliothek. Dieses Symbol ist mit einer ActionScript Klasse verknüpft, die von der Basisklase `flash.display.SimpleButton` abgeleitet ist. Die jeweiligen Instanzen wurden mit eindeutigen Instanznamen versehen, um diese im ActionScript Quellcode referenzieren zu können. So war es möglich, eine Schaltfläche mit einem EventListener zu verknüpfen.

Weiterhin wurden in Flash CS3 die Textfelder für die Bezeichnungen der Softkeys erzeugt, auf dem Display der Handyoberfläche positioniert und mit Instanznamen versehen. Somit konnten diese im Quellcode mit variablem Text belegt werden.

Damit im ActionScript Quelltext auch das Display des Mobiltelefons mit verschiedenen Inhalten belegt werden kann, wurde in der Flash-Bibliothek ein Symbol `Screen` mit einem dazugehörigen Bild erzeugt und mit der gleichnamigen ActionScript Klasse verknüpft, die von der Basisklasse `flash.display.MovieClip` abgeleitet ist. Von diesem Symbol wurde wiederum eine Instanz mit einem Instanznamen erzeugt, auf den im Quelltext zugegriffen werden kann.

Ebenso wurden zu allen angezeigten Grafiken wie z.B. der Karte, den Menüansichten oder den POIs eigene Symbole erzeugt, die mit einer eigenen Klasse verknüpft wurden und von `flash.display.MovieClip` abgeleitet sind. Für die einzelnen Menüpunkte wurde eine eigene Basisklasse geschrieben, von der die verschiedenen Sym-

bole abgeleitet wurden. Dies war nötig, weil in jedem Menüpunkt verschiedene Variablen enthalten sein müssen, beispielsweise, zu welchem Menü die Punkte gehören und auf was sie verlinken. Ebenso beinhalten die verschiedenen Menüansichten Variablen. Die Menüansichten sollten jedoch von ihrer grafischen Darstellung entkoppelt sein, da es nicht vom Menü, sondern vom einzelnen Menüpunkt abhängt, ob die Menüansicht ein Submenü inkludiert oder nicht. Aus diesem Grund wurden die verschiedenen Menüansichten nur von der Basisklasse `flash.display.MovieClip` abgeleitet.

8.2 Die Softwarearchitektur

Nachdem im vorigen Abschnitt die Interaktion zwischen Flash CS3 und dem Quellcode von Actionscript vorgestellt wurde, soll nun auf die Softwarearchitektur des Quelltextes eingegangen werden. Diese ist folgend in einem vereinfachten UML-Diagramm visualisiert:

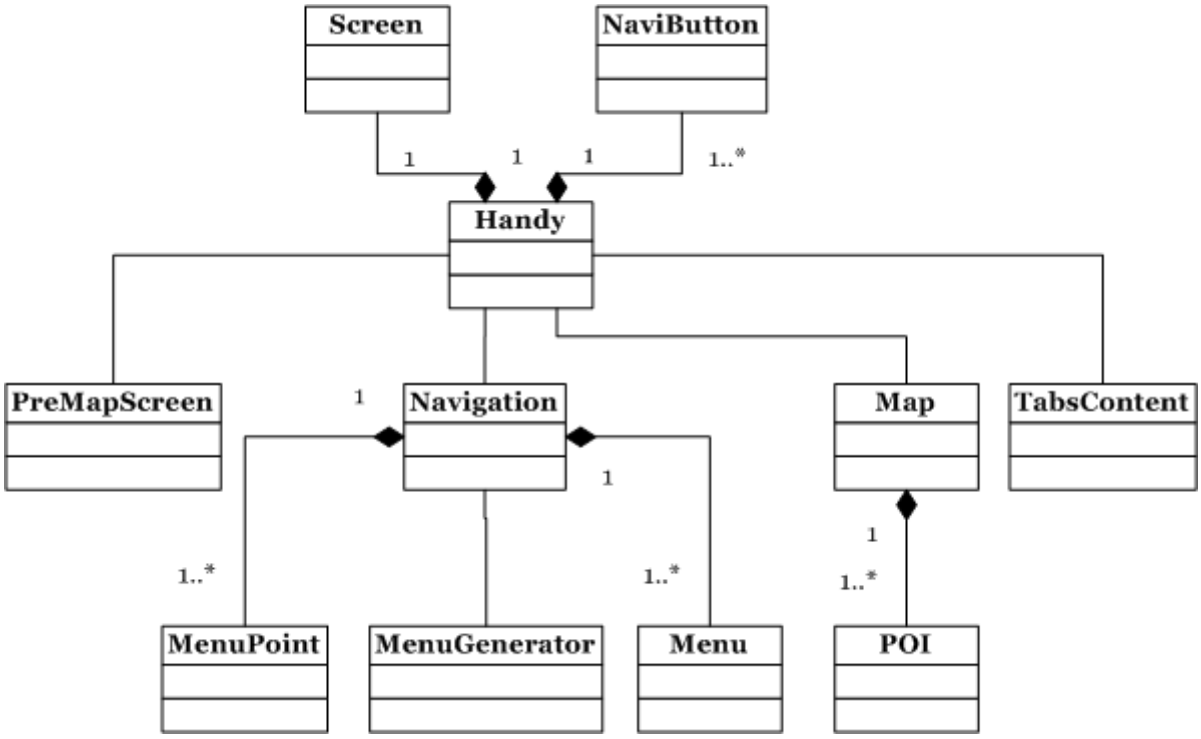

Abbildung 5: Vereinfachtes UML-Diagramm der Softwarearchitektur

In den folgenden Kapiteln sollen nun die verschiedenen zentralen Klassen kurz beschrieben werden.

8.2.1 Die Klasse Handy

Wie dem UML-Diagramm aus Abbildung 5 entnommen werden kann, ist die Klasse `Handy` die zentrale Klasse des Prototyps. Diese ist ebenso die Dokumentenklasse der Flash-Datei und wird daher beim Kompilieren instanziiert. Diese Klasse organisiert das Display des Prototyps und die EventListener der Schaltflächen zur Navigation. Zusätzlich dazu verwaltet diese die verschiedenen Anzeigemodi `Navigation`, `PreMapScreen`, `Map` und `TabsContent`. Von diesen einzelnen Anzeigemodi kann über die Klasse `Handy` auf die Textfelder zugegriffen und ein anderer Modus geladen werden. Somit benötigt jeder Anzeigemodus ebenso eine Referenz auf das Objekt der Klasse `Handy`.

Falls in einen anderen Modus gewechselt wird, werden die Textfelder mit neuen Inhalten belegt, die EventListener der Schaltflächen neu vergeben und der neue Inhalt eingeblendet. Die Eingabeevents werden so von der Klasse Handy an die Methode `navigationHandler(event:Event)` des Objektes des jeweiligen Anzeigemodus weitergegeben. Somit verfügt dieses über die Kontrolle der Navigation. Dies ist notwendig, da in jedem Modus eine andere Aktion durch die Buttons ausgelöst werden muss. Beispielsweise soll in der Menüansicht bei Betätigen des oberen Pfeils des Steuerkreuzes das Submenü aufgerufen werden, während in der Kartenansicht die Karte nach oben gescrollt werden soll.

Die Anzeigemodi des Prototyps lassen sich in zwei Bereiche aufteilen: Inhalt und Navigation. Die Navigation beinhaltet die verschiedenen Menüansichten mitsamt ihrer Menüpunkte und deren Verweise auf den jeweiligen Inhalt. Dieser besteht aus den verschiedenen Anzeigemodi `PreMapScreen`, `Map` und `TabsContent`. Aus diesem Grund wird der Anzeigemodus der Navigation direkt beim Programmstart geladen. Die dazugehörige Klasse wird im folgenden Kapitel beschrieben.

8.2.2 Die Klasse Navigation

Wie schon im vorigen Kapitel angesprochen wurde, beinhaltet die Klasse Navigation die verschiedenen Menüs, deren Menüpunkte und die Abhandlung der Navigation in diesen. So werden in einem ersten Schritt in dieser Klasse die verschiedenen Menüs und Menüpunkte generiert. Dies geschieht in der Klasse `MenuGenerator`, von der eine Instanz erzeugt und ein Verweis auf das aktuelle Objekt der Klasse Navigation übergeben wird. In dieser Klasse sind somit die verschiedenen Namen und Verlin-

kungen der Menüs gekapselt. Nachdem so die Menüs und die dazugehörigen Menüpunkte generiert und in ein Array übertragen wurden, werden von den verschiedenen Arten der Menüs (Menü der Ebene 1 mit/ohne Submenü, Menü der Ebene 2) Instanzen erzeugt, die in der Flash Bibliothek mit den jeweiligen Bildern verknüpft sind. Anschließend wird die Laufzeitvariable `currentMenuIndex` auf den Index des obersten Menüs gesetzt und dieses angezeigt.

Falls der Anwender eine der Schaltflächen auf der Handyoberfläche betätigt, werden die Events an die zentrale Methode `navigationHandler(event:Event)` weitergereicht. Diese ruft je nach Art des Events verschiedene Funktionen auf. Wird der obere Pfeil des Steuerkreuzes betätigt, wird zunächst geprüft, ob ein Submenü zu dem aktuellen Menüpunkt existiert. Wenn dem so ist, wird die Variable `currentMenuIndex` auf das Submenü gesetzt und dieses angezeigt. Dies geschieht ebenso, wenn der mittlere Softkey betätigt wurde und ein Submenü existiert. Falls dies nicht existiert, wird der Inhalt aufgerufen, auf den der Menüpunkt verlinkt. Bei Betätigen des unteren Pfeils des Steuerkreuzes wird analog dazu kontrolliert, ob ein Elternmenü existiert und bei erfolgreicher Prüfung der `currentMenuIndex` auf dessen Index gesetzt und dieses aufgerufen. Falls der linke oder rechte Pfeil des Steuerkreuzes vom Anwender betätigt wurde, werden die Menüpunkte des aktuellen Menüs in einem Karussell verschoben. Dazu wird der Fokus des aktuellen Menüpunktes im Menü dekrementiert bzw. inkrementiert und anschließend der Menüstand aktualisiert.

Im Folgenden soll der zentrale Algorithmus vorgestellt werden, der die aktuell angezeigten Menüpunkte eines Menüs setzt. Hierzu muss erwähnt werden, dass eine Menüebene eine beliebige Anzahl von Menüpunkten enthalten kann und abhängig vom aktuellen Fokus nur fünf davon dargestellt werden sollen. Dazu werden die zu präsentierenden Menüpunkte von diesem Algorithmus in ein Array geschrieben und an die Methode zurückgegeben, die das aktuelle Menü anzeigen soll.

```
var visibleElementsArr:Array = new Array();
for (var i:uint = 0; i < 5; i++) {
      j = i - 2;
      visibleElementsArr[i] = menus[menuIndex].currentFocus + j;
      if (visibleElementsArr[i] < 0) {
            visibleElementsArr[i] += elementCount;
      }
      else if (visibleElementsArr[i] >= elementCount) {
            visibleElementsArr[i] -= elementCount;
      }
}
```

Codebeispiel 1: Der Algorithmus zur Berechnung der dargestellten Menüpunkte

Abhängig vom aktuell angezeigten Menüpunkt (`currentFocus`) werden die beiden vorigen und folgenden Menüpunkte angezeigt. Falls dessen Index nicht innerhalb des Arrays liegt, muss die Länge des Arrays von diesem subtrahiert bzw. addiert werden, um die Anzeige als ein Karussell zu ermöglichen.

Nachdem von diesem Algorithmus die angezeigten Elemente ermittelt wurden, werden diese auf das Menü gezeichnet. Um die Fish-Eye Optik zu erzielen, werden die verschiedenen Elemente abhängig von ihrer Lage automatisch skaliert und in das Menü integriert. Falls ein Submenü oder Elternmenü existiert, werden die darin enthaltenen Elemente ebenfalls mit diesem Algorithmus berechnet, in ihrer Größe skaliert und in die jeweilige Ebene gezeichnet.

Für jeden Menüpunkt muss somit berechnet werden, ob zu diesem ein Submenü und ein Elternmenü existiert oder nicht. Um die korrekte Menüansicht anzuzeigen, muss weiterhin die Ebene des aktuellen Menüs überprüft werden. Über die Instanz des aktuell fokussierten Menüpunktes kann auf den Index des dazugehörigen Submenüs zugegriffen werden. Falls dieser den Wert -1 hat, existiert kein Submenü und der Menüpunkt verlinkt somit auf einen Inhalt. Auf die Ebenenbezeichnung und den Index des Elternmenüs kann über die Instanz des aktuellen Menüs zugegriffen werden. Falls kein Elternmenü existiert, hat dieser Index ebenfalls den Wert -1.

8.2.3 Die Klasse PreMapScreen

Die Klasse PreMapScreen enthält die Anzeige der Checkboxen, die vor der Karte dargestellt werden (siehe 7.2).

Um dieses zu realisieren, wurden die verschiedenen Checkboxen in ein Array zusammengefasst und mittels der Laufzeitvariablen `currentCheckboxIndex` angesprochen. Falls der obere oder untere Pfeil des Steuerkreuzes des Mobiltelefons betätigt wurde, wird diese Variable dekrementiert bzw. inkrementiert. Die aktuelle Checkbox wird dann von den anderen hervorgehoben. Durch Drücken des mittleren Softkeys wird die aktuell ausgewählte Checkbox selektiert oder deselektiert. Der rechte Softkey ruft über die Referenz auf die Instanz der Klasse `Handy` die Kartendarstellung auf.

8.2.4 Die Klasse Map

Die Klasse Map besteht aus einer Kartendarstellung und mehreren Elementen, die auf dieser Karte angezeigt werden. Diese Elemente lassen sich in zwei Arten unterscheiden: Variable Elemente, die mit der Karte verschoben werden sollen und Elemente, die einen fixen Platz auf der Anzeige einnehmen. Zu den variablen Elementen gehören die POIs und die dazugehörigen Sprechblasen. Das Fadenkreuz, die Legende, die Zoomleiste und das Längenmaß sind dahingegen fixe Elemente. Um diese verschiedenen Elemente zu gruppieren, wurden zwei Container in die Klasse integriert. Der `mapContainer` enthält die variablen Elemente und wurde aus diesem Grund mit dem Funktionsaufruf `mapObject.addChild(mapContainer)` der Karte als Kind hinzugefügt. Dahingegen wurde der `uiContainer`, der die fixen Elemente beinhaltet, dem Anzeigemodus selber als Kind hinzugefügt.

Die beispielhaften POIs des Prototyps sind in einem Array gespeichert. Da zu jedem POI eine eigene Sprechblase gehört, in der zentrale Informationen über diesen POI stehen, wurden die POIs als eigene Klasse realisiert. In dieser wird im Konstruktor die zugehörige Sprechblase erzeugt, positioniert und formatiert und dieser der übergebene Text zugewiesen.

Durch die vier verschiedenen Pfeile des Steuerkreuzes wird die Karte um jeweils 10 Pixel verschoben, indem der x- bzw. y-Wert des Kartenobjektes dementsprechend verändert wird. Durch den mittleren Softkey wird in der Kartendarstellung die Zoom-Funktion aufgerufen. Falls sich das Fadenkreuz über einem POI befindet und so eine Sprechblase erscheint wird ein neuer EventListener benötigt, damit durch den mittleren Softkey statt der Zoom-Funktion die Seite des POIs angezeigt wird.

In der Zoom-Funktion werden eigene EventListener für den oberen und unteren Pfeil des Steuerkreuzes benötigt, damit der Nutzer so die verschiedenen Zoomstufen regeln kann. Diese ist jedoch nicht als eigene Klasse realisiert, da der mittlere Softkey weiterhin mit der Kartendarstellung verbunden ist, um die Zoom-Funktion ein- und auszublenden.

Durch den rechten Softkey wird der Navigationsmodus durch die Referenz auf das Handyobjekt wieder aufgerufen.

8.2.5 Die Klasse TabsContent

Um in der Klasse TabsContent die verschiedenen Register mitsamt ihrer eigenen Textfelder anzeigen zu können, wird ein dreidimensionales Array benötigt. In der ersten Dimension verlinkt ein Arrayfeld auf ein neues Array. In diesem ist im ersten Feld die visuelle Ansicht des Registers gespeichert und im zweiten Feld wird wiederum ein weiteres Array referenziert, in dem sich die zugehörigen Textfelder befinden.

Falls der linke oder rechte Pfeil des Steuerkreuzes betätigt wurde, wird ein neues Register angezeigt, indem die Laufzeitvariable `currentTabScreenIndex` dekrementiert bzw. inkrementiert und die neue visuelle Ansicht geladen wird. Bevor dies geschieht, wird zunächst geprüft, ob die gewünschte Veränderung überhaupt möglich ist. Falls eine neue Registerkarte angezeigt wird, wird die Variable `currentTextfieldIndex` auf 0 gesetzt, um jeweils das oberste Textfeld der Registerkarte hervorzuheben. Wenn nun der Nutzer auf den oberen oder unteren Pfeil des Steuerkreuzes drückt, wird diese Variable dekrementiert oder inkrementiert, falls dies möglich ist. Anschließend wird das Textfeld mit dem neuen Index hervorgehoben. Der rechte Softkey ruft wiederum die Navigation des Prototyps auf.

9 Fazit

Trotz fehlender Vorkenntnisse in der Programmiersprache ActionScript und der Entwicklungsumgebung von Flash konnten die zentralen Gestaltungsvorschläge des Informations- und Interaktionsdesigns in einen interaktiven Prototyp umgesetzt werden. Die größte Herausforderung stellte hierbei neben der Einarbeitung in die Programmiersprache ActionScript die Entwicklungsumgebung *Flashdevelop* dar. Die Umstände, dass kein Debugger existiert, kein Code-Refactoring möglich ist und die Programmfehler nicht sofort angezeigt werden, hat viel Zeit bei der Entwicklung dieses Prototyps gekostet.

Bei der Umsetzung spielte die Qualität des Quelltextes keine große Rolle, da aufgrund des engen Zeitrahmens möglichst schnell Ergebnisse erzielt werden mussten und die Entwicklung eines Prototyps generell wenig Aufwand erfordern sollte (siehe 4.2). Trotzdem wurde darauf Wert gelegt, dass der Prototyp prinzipiell erweiterbar bleibt. Dies ist durch die ausgelagerte Generierung der Menüs inklusive deren Verlinkung und durch die verschiedenen voneinander gekapselten Anzeigemodi möglich. Nachteilig ist dahingegen beispielsweise, dass viele Variablen als public deklariert sind und alle Bezeichnungen im Quelltext programmiert und nicht in ein XML-Dokument ausgelagert sind. Mit einer professionellen Entwicklungsumgebung können diese Schwachstellen jedoch problemlos beseitigt werden. Anzumerken ist, dass die visuelle Feingestaltung nicht Bestandteil dieses Prototyps ist und aus diesem Grund die Icons nur grob skizziert sind und fast der gesamte Prototyp in Graustufen gehalten ist. Dadurch ist ein Hinweis auf die benötigten Helligkeitsunterschiede gegeben, während in der Farbwahl Freiheiten gelassen werden.

Durch dieses Projekt und die Bachelor-Thesis existiert nun eine Basis, um das Konzept der entworfenen Benutzungsschnittstelle zu evaluieren und in einer weiteren Iteration zu optimieren.

Quellen- und Literaturverzeichnis

[1] Nolte, S.: Entwicklung eines User Interface Konzeptes für eine mobile Applikation eines Location Based Social Networks auf Basis des benutzerzentrierten Gestaltungsprozesses; Hochschule der Medien – Bachelor-Thesis im Studiengang Medieninformatik WS07/08; Stuttgart; 2008