Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Michael Pawlik

Mobile Story Board - Entwicklung einer Android App zur Vertriebsunterstützung von Immobilien

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Michael Pawlik

Mobile Story Board - Entwicklung einer Android App zur Vertriebsunterstützung von Immobilien

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Bachelor of Science Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Birgit Wendholt, Prof. Dr. Olaf Zukunft

Eingereicht am: 26. Juli 2012

Michael Pawlik

Thema der Arbeit
Mobile Story Board - Entwicklung einer Android App zur Vertriebsunterstützung von Immobilien

Stichworte
mobile, storyboard, android, location, locationbased

Kurzzusammenfassung
Dieses Dokument beschäftigt sich mit dem Entwurf einer mobilen Applikation zur Unterstützung von Immobilienberatern bei der Erstellung Immobilienrepräsentierender Dokumente. Als Alternative zu einem Exposê wird hier der Begriff eines Storyboards eingeführt, unter dem man die Präsentation eines Objekts und dessen Umfeld versteht. Mit Hilfe von Locationbased Services sollen Objekt- und Umgebungsinformationen zu einer Story zusammengefügt werden, die durch die von mobilen Endgeräten unterstützen Medien erweitert werden kann.
Michael Pawlik

Title of the paper
Mobile Story Board - Development of an sales supporting android app of real estate

Keywords
mobile, storyboard, android, location, locationbased

Abstract
This document's purpose is the design of a mobile application for the support of real estate consultants in the preparation of real estate representing documents. As an alternative to an exposê, this document introduces the concept of storyboards, which is understood as a presentation of a property and its vicinity. With the help of locationbased services, informations about the porperty and its vicinity are combined into a story. The story can be extended by media types, which are supported by mobile devices.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Für die Vermarktung einer Immobilie ist es dringend erforderlich, diese ausreichend und repräsentativ durch Texte, Bilder und Videos darzustellen, um bei den potentiellen Käufern oder Mietern schon vor einer Besichtigung das Interesse zu wecken. Neben Innen- und Außenaufnahmen, dem Grundriss (falls vorhanden) und Beschreibungen der Ausstattung einer Immobilie werden oft noch Informationen über die Umgebung angegeben. Dazu gehören Informationen über Schulen, Kindergärten, Ärzte, Krankenhäuser und andere "Point of Interrest"(POI). Sind alle Informationen zusammengetragen, werden diese entweder online oder manuell in einem Exposé zusammengefasst, das den etwaigen Käufern oder Mietern als Informationsgrundlage über die Immobilie dient. Die bisher genutzte automatische Generierung eines Exposés nutzt die in die Maklersoftware hochgeladenen Bilder und Texte und verbindet diese mit Informationen wie Preis, Größe und Adresse. Um dem Kunden seine neue Immobilie näher zu bringen reicht ein Exposê aber meist nicht aus. Hier kommt das Storyboard ins Spiel. Es beinhaltet nicht nur Bilder und Informationen über die Immobilie selbst, sondern zeigt durch zahlreiche Informationen und Medien die Umgebung und das Leben in der Nähe der Immobilie. Dem Immobilienberater sollen alle Informationen zur Verfügung gestellt werden, die er benötigt um seinen Kunden das Gefühl zu vermitteln, sie würden schon lange in dieser Umgebung leben. Neben dem hohen manuellen Aufwand, der durch die Recherche und Zusammenführung aller Informationen ensteht, bleibt weitere manuelle Arbeit durch das Hochladen eines manuell erstellten Exposés oder dem Hochladen aller Bilder und Texte in die Maklersoftware, um ein Exposé automatisch erstellen zu lassen. Dieser manuelle Aufwand ist unnötig denn durch moderne Techniken wie Locationbased Services, die sich der aktuellen Position seines Nutzers bedienen, um standortbezogene Daten bereitzustellen, erübrigt sich die manuelle Recherche nach Point of Intrest's (POI's) in der Umgebung. Das manuelle Hochladen der einzelnen Information oder eines erstellten Exposés kann automatisiert werden. Diese Automatisierung sowie eine komplexere Darstellung von Immobilien ist das Ziel der in dieser Arbeit entstehenden Anwendung.

1.1 Ziele

Ziel der Arbeit soll der Entwurf einer mobilen Applikation sein, welche es einem Immobilienberater ermöglicht, die ihm zur Verfügung stehenden Daten über eine Immobilie zusammen mit Umgebungsinformationen zu einer Story der Immobilie zusammenzustellen. Hierbei sollen möglichst viele der Medien zum Einsatz kommen, die durch die Laufzeitumgebung verfügbar gemacht werden. Bei der Laufzeitumgebung sollte Wert auf eine hohe Verbreitung gelegt werden und es sollte der OpenSource Gedanke durch diese vertreten werden. Die Laufzeitumgebung der Applikation soll möglichst weit verbreitet sein. Der OpenSource Gedanke soll unterstützt werden. Umgebungsinformationen sollen in Abhängigkeit der aktuellen Position des Benutzers über Locationbased Services wie Google Places eingeholt und in die Story eingearbeitet werden. Bei dem Entwurf ist darauf zu achten, dass eine Erweiterbarkeit der Locationbased Service Provider jederzeit möglich ist, so dass solche leicht nachträglich in die Applikation integriert werden können. Als Proof Of Concept ist die prototypische Implementierung des Entwurfs Teil der Arbeit.

1.2 Gliederung

Das Kapitel 2 beschäftigt sich mit vergleichbaren Arbeiten um einen Blick auf den aktuellen Stand der Technik zu geben.
Im Kapitel 3 werden die Ziele und Anforderungen anhand des Ist- und Ziel-Zustands abgeleitet. Anhand dieser werden im weiteren Verlauf die Akteure, Entitäten und Anwendungsfälle definiert. Desweiteren wird der Begriff Locationbased Service erläutert.
Anhand der im Kapitel 3 durchgeführten Analyse, werden im Kapitel 4 die Anforderungen an Soft- und Hardware definiert. Die im Entwurf verwendeten Designpatterns werden erklärt und in Bezug zur Arbeit gestellt. Zudem werden dort Konzepte der Zielplattform erläutert. Nach dem Komponentenschnitt, welcher eine Übersicht über Aufgabenverteilung im Entwurf geben soll, werden interne Abläufe durch Klassendiagramme, Sequenzdiagramme und Screenshots klar gemacht. Das letzte Kapitel 5 fasst die Arbeit zusammen und soll einen Ausblick auf mögliche Erweiterungen geben.

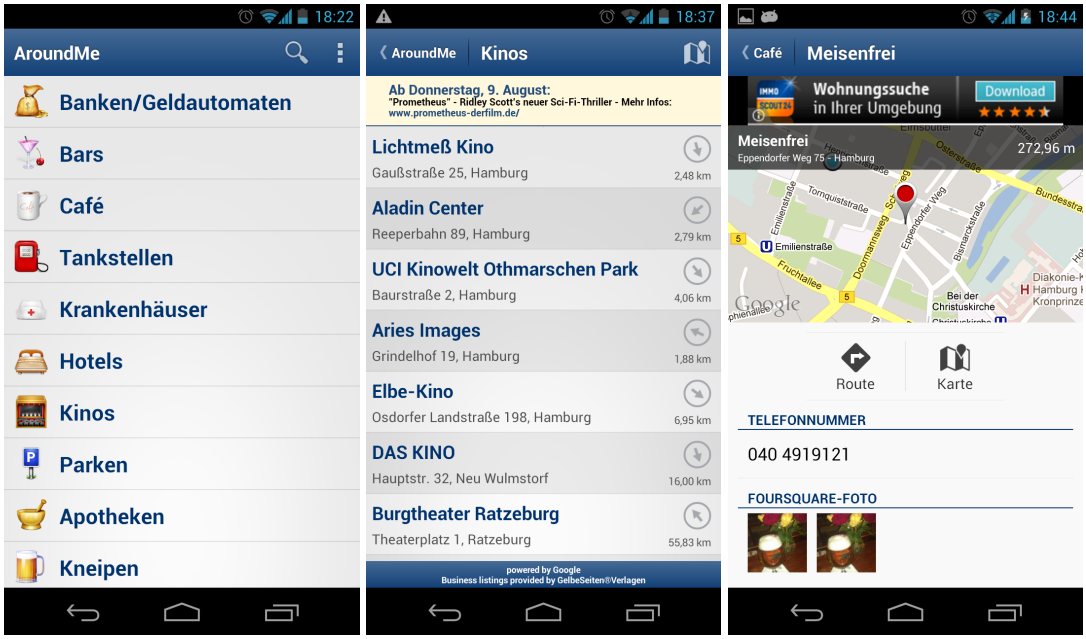
2 Vergleichbare Arbeiten

Um einen Einblick auf den aktuellen Stand der Technik sowie vergleichbare Arbeiten zu erhalten, befasst sich dieses Kapitel mit sich zur Zeit in der Entwicklung befindlichen oder schon vorhandenen Applikationen. Es wird sich hierbei um Applikationen handeln, die sich auf mobilen Endgeräten befinden und Daten anhand ihrer aktuellen Position durch Umgebungsinformationen anreichern.

2.1 AroundMe

Mit der Applikation AroundMe ist es möglich in seinem direkten Umfeld nach verschiedensten POIs zu suchen. Eine solche Suche kann entweder über Kategorien oder über die Eingabe eines

Abbildung 2.1: Screenshots: Aroundme App



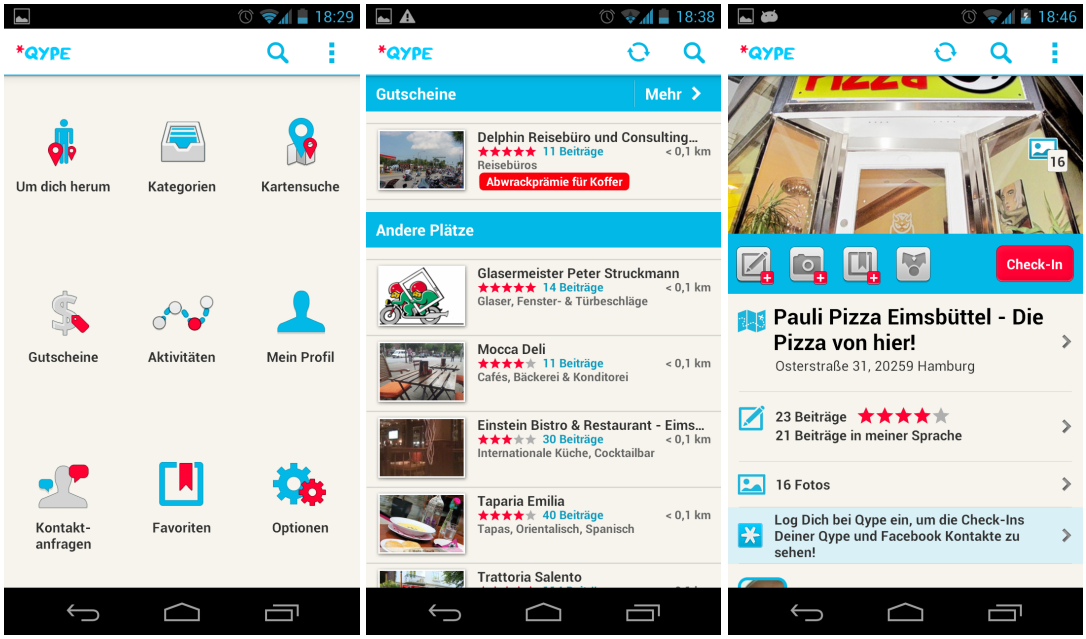
Suchbegriffs geschehen. Die Ergebnisliste beinhaltet neben den Namen der POIs auch deren

Entfernung zur aktuellen Position sowie deren ungefähren Richtung. In der Detailansicht eines POI's erhält man neben einer Karte, in der die Position des POI's angezeigt wird, auch Informationen über Telefonnummern. Es ist zudem möglich sich über eine externe Kartenapplikation oder über die Internetseite von Google Maps eine Route von der aktuellen Position bis zum POI ausgeben zu lassen. Bildinformationen werden in der Applikation durch den externen Dienstleiter Foursquare bereitgestellt. Dieser bietet dem Benutzer ebenfalls Kommentare bzw. Tipps von anderen Benutzern an.

2.2 Qype

Wie in der AroundMe Applikation ist es auch in der Qype App möglich, POIs aus der Umgebung zu suchen oder nach Kategorien anzeigen zu lassen. Hinzu kommt die Möglichkeit der Kartensuche. Mit dieser navigiert man durch Scrollen und Pinchen (Heraus- und Hereinzoo-

Abbildung 2.2: Screenshots: Qype App



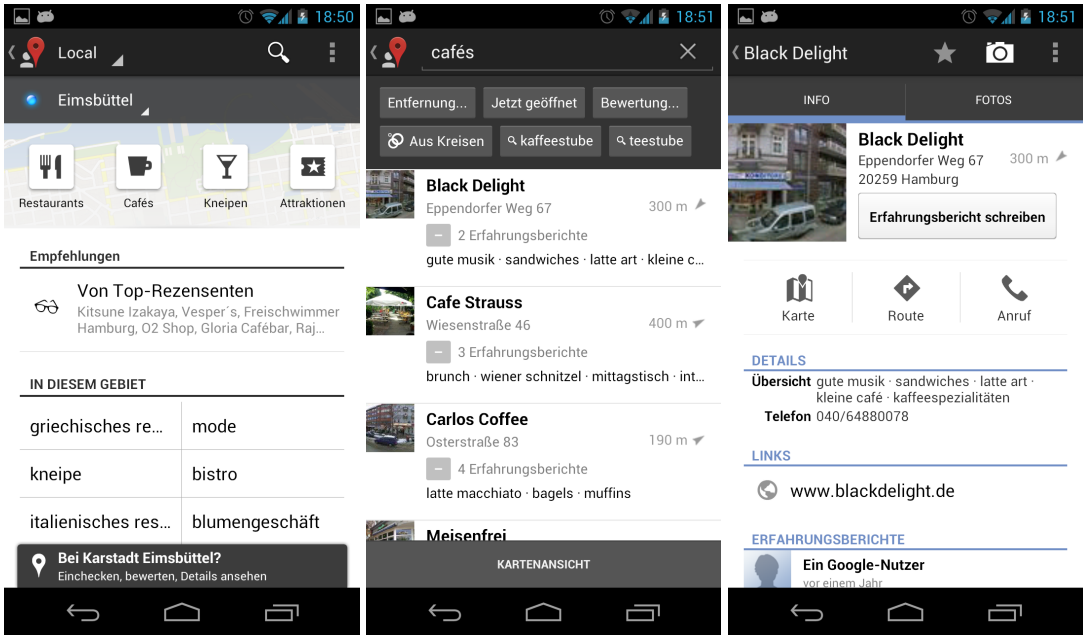
men durch Fingergesten) zu einer beliebigen Stelle und kann den angezeigten Kartenausschnitt jederzeit mit POI Informationen aufwerten. Hierbei werden kleine Symbole an den Stellen angezeigt, an denen sich die POIs befinden. Neben diesen Funktionen bietet die Qype App außerdem ein soziales Netzwerk, über welches man in seinem Benutzerkonto Favoriten ver-

walten, neue POIs zum Netzwerk hinzufügen oder die Aktivitäten seiner Freunde verfolgen kann.

2.3 Google Local

Google bietet in seiner 'Local' Applikation ähnliche Dienstleistungen an wie die beiden Applikationen zuvor. Wie in der Qype App ist es auch in dieser Applikation möglich, sich Informationen auch zu entfernten liegenden Positionen anzeigen zu lassen. Die Favoriten-

Abbildung 2.3: Screenshots: Google Maps App



verwaltung erfolgt hier nicht über das Merken von POIs sondern über das Speichern von Suchen. Wie Qype ist auch diese Applikation an ein soziales Netzwerk angebunden - Google Plus. In der Detailansicht der einzelnen POIs kann man sich neben Kommentaren aus dem hauseigenen Netzwerk auch Kommentare aus Qype und anderen Netzwerken ansehen.

2.4 Zusammenfassung

Wie man in den zuvor beschriebenen Applikationen sehen kann, handelt es sich bei diesen eher um Suchapplikationen. Es geht also mehr darum eine Lokal oder Kino zu finden als einen Ort oder eine Umgebung zu beschreiben. Die Bezugsgröße aller Applikationen ist der Point of Intrest (POI). Er dient als geografische Referenz, welche durch Medien, Kommentare oder Bewertungen angereichert werden kann. Im folgenden Kapitel sollen nun im Detail die bisher erwähnten Eigenschaften des MobileStoryboards erarbeitet werden.

3 Analyse

Um eine Applikation erfolgreich zu konzipieren, bedarf es einer Analyse ihrer Ziele und Anforderungen. Es ist wichtig zu wissen, was durch diese Applikation bewerkstelligt werden soll, welche Akteure bei der Erfüllung ihrer Ziele mitarbeiten und welche Ressourcen hierfür benötigt werden. Um konkrete Anforderungen stellen zu können, sollte zuerst definiert werden, wie die aktuelle Situation aussieht und was es zu verbessern gilt. Hierzu wird im Folgenden eine Beschreibung des Ist-Zustands gegeben, welcher Aufschlüsse über die unterschiedlichen Entitäten, Akteure und Anforderungen an die zu entwickelnde Applikation geben soll. Nach dieser Erörterung wird auf die herausgefilterten Entitäten und Akteure eingegangen, die in den darauf folgenden funktionalen Anforderungen benötigt werden.

3.1 Ist-Zustand

Wie im Kapitel 1 beschrieben,werden Informationen über die Umgebung einer Immobilie zurzeit wenn überhaupt über eine manuelle Suche gefunden. Bilder und Videos werden mit Hilfe von digitalen Kameras aufgenommen und dann im Büro zur weiteren Bearbeitung auf den Computer übertragen. Alle objektrelevanten Daten werden in der hauseigenen Software gespeichert und die gewünschten Bilder in dieses System hochgeladen. Ein Exposé kann hier dann durch die verwendete Software automatisch generiert oder als manuell erstelltes PDF hochgeladen werden.

3.2 Ziel-Zustand

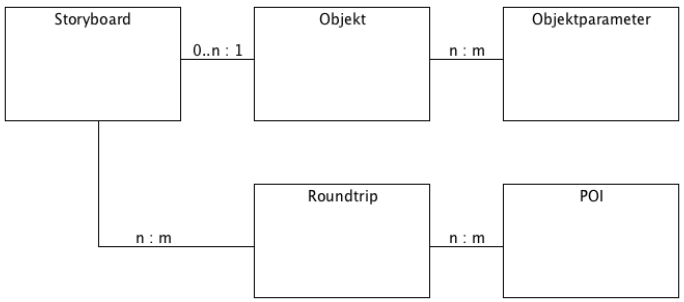
Über die in dieser Arbeit zu entwickelnde Proof of Concept Applikation soll es möglich sein, Informationen über ein Objekt aufnehmen zu können. Diese Informationen sollen über externe Dienste mit Umgebungsinformationen angereichert werden können. Als Umgebungsinformationen gelten hier besondere Orte, Gaststätten, Schulen, Theater oder Ähnliches. Zusätzlich ist die Möglichkeit erforderlich Medien zu dem Objekt hinzuzufügen. Um das Weiterverarbeiten der zusammengetragenen Informationen zu ermöglichen, sollten diese auch ex- und importiert werden können. Hierdurch erhält der Immobilienberater die Möglichkeit, seine auf dem

mobilen Endgerät erstellte Komposition von Objekt- und Umgebungsinformation, auf seinem Computer zu editieren oder seinen Kollegen zur Verfügung zu stellen.

3.3 Akteure und Entitäten

Aus der Ist-Zustands Analyse geht hervor, dass es sich bei dem einzigen Akteur um den Immobilienberater handelt, der mithilfe der zu entwickelenden Applikation seine Aufgaben effizienter erledigen soll. Weiterhin lässt sich eine Immobilie als Entität bestimmen, die, im folgenden Objekt genannt, repräsentativ Informationen über die Größe, die Adresse, Ausstattung oder Ähnliches beinhaltet. Um die Parametrisierung eines Objekts variabel zu halten, werden Informationen, die über die Basisinformation der Lokalität hinausgehen, in Objektparametern gehalten. Somit kann die Applikation während ihrer Anwendung erweitert und auf die Bedürfnisse ihres Benutzers angepasst werden. Um auch die Parameter variabel zu halten, sollten diese einen Typ (z.B.: Zahl, Text, oder Datum) sowie einen Wert haben. Um die Ziele dieser Applikation umsetzen zu können werden weitere Entitäten benötigt, die im folgenden beschrieben werden. Point of Intrests (POI's) sind Repräsentationen von Orten oder Gebäuden in der echten Welt. Sie können neben ihrer Adress- und GEO- Information

Abbildung 3.1: Darstellung der Entitäten



auch Telefonnummern oder Bilder enthalten. Als Beispiel seien hier Schulen, Theater, Bars, Einkaufsläden oder Aussichtspunkte zu nennen. Storyboards repräsentieren Objektpräsentationen. Sie sind eine Alternative zu Exposés und sind mit dem zu präsentierenden Objekt verknüpft. Die Anreicherung mit Umgebungsinformationen und Medien geschieht durch POIs. Um den Aufwand der Verknüpfung von Storyboards und POIs für Objekte im gleichen Stadtteil zu reduzieren, werden die POIs nicht direkt mit den Storyboards verknüpft, sondern

zunächst in Roundtrips gebunden. Ein Roundtrip fasst mehrere POIs zusammen, um einen bestimmten Aspekt aus einem Stadtteil einzufangen. Beispielsweise könnte es für den Stadtteil Eimsbüttel einen Roundtrip für Familien mit Schulen und Kindergärten sowie einen Roundtrip für Studenten mit Bibliotheken, Discotheken und Bars geben. Storyboards für Objekte aus Eimsbüttel können nun Roundtrips aus der gleichen Gegend benutzen und die verknüpften Objekte für ein unterschiedliches Kundenklientel aufbereiten. Jedes Storyboard ist mit dem Objekt verknüpft, das es präsentiert und hat beliebig viele Roundtrips. Um an die Informationen der POIs zu gelangen, wird eine Anbindung an sogenannte LocationBasedServices benötigt, deren Funktion im Folgenden beschrieben wird.

3.4 Locationbased Services

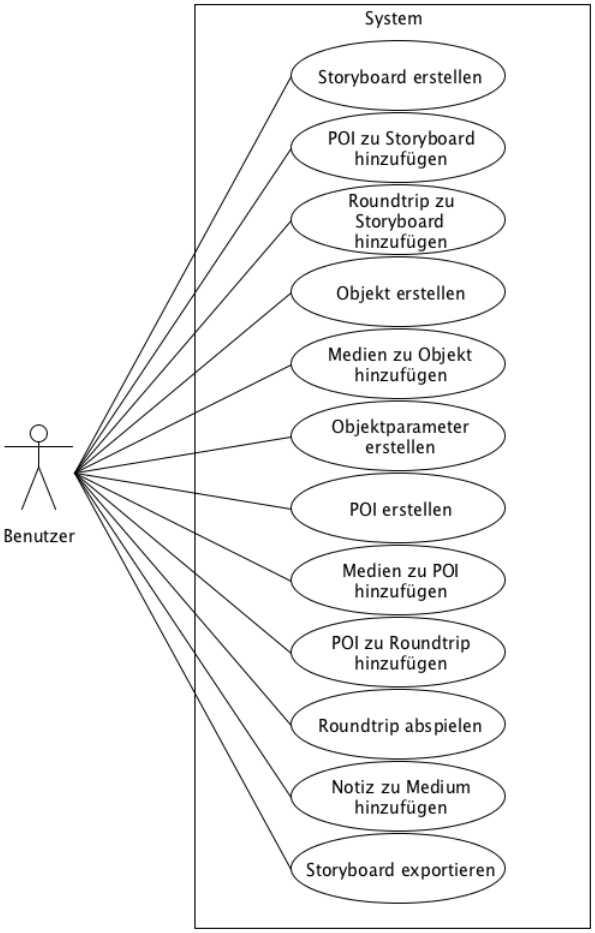
Unter Locationbased Services (LBS) versteht man Dienste, die in Abhängigkeit von einer geografischen Position, Informationen zu dieser und ihrer Umgebung bieten. Dies können neben speziellen Orten und interessanten Gebäuden auch Wetter- oder Verkehrsinformationen sein. In Bezug auf diese Arbeit und in Anbetracht der in den vorhergehenden Kapiteln beschriebenen POIs, sind besonders die Provider interessant, die Orts- und Gebäudeinformationen anzubieten haben. Die Kommunikation mit solchen Diensten erfolgt meist über über ein Applikations Interface, welches über das HTTP Protokoll angeboten wird. Über diese API werden die Anfragen dann inf Form einer URL angegeben und die Ergebnisse in Form von XML oder JSON Daten zurückgeschickt. Bei den meisten Service Providern ist es weiterhin nötig bei allen Anfragen einen so genannten API-Key mit anzugeben, welcher als eindeutige Kennung des Klienten gilt. Über diesen Key kann der Service Provider dann zum Beispiel die maximale Anzahl von Anfragen pro Minute an den Service limitieren. Die LBS Provider bieten über die API meist unterschiedliche Arten von Anfragen an. Das Befragen nach POIs zu einer Lokation findet meist separiert von Detailanfragen zu einem POI des Services statt. Bei der letzteren Anfrage ist es nötig, einen eindeutigen Schlüssel für den gewünschten POI mit anzugeben, wobei sich die Eindeutigkeit dieses Schlüssels auf einen Service Provider beschränkt. Dies bedeutet, dass ein POI mit Schlüssel X aus Provider A nicht dem POI aus Provider B mit Schlüssel X entsprechen muss.

3.5 Funktionale Anforderungen

Durch die im Kapitel 3.3 durchgeführte Analyse der Akteure und Entitäten, ist es nun möglich, funktionale Anforderungen zu definieren, die im späteren Verlauf der Arbeit den Funktionsumfang der Prototyp Applikation bestimmen werden. Da es sich bei den Akteuren ausschließlich um den Immobilienberater bzw. Benutzer handelt, werden sich im Folgenden alle Anwendungsfälle auf diesen beziehen.

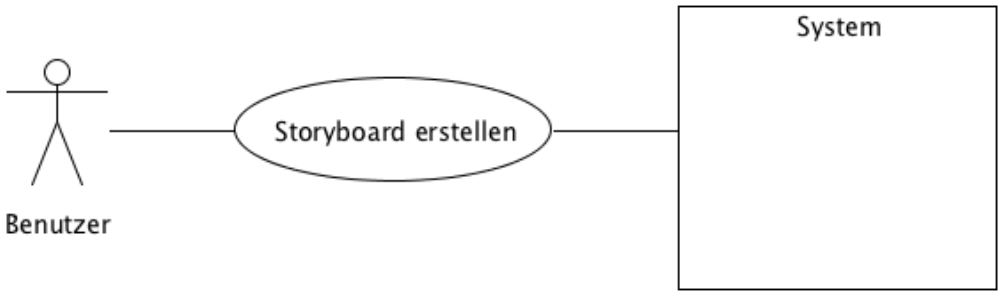
3.5.1 Anwendungsfall Überblick

Abbildung 3.2: Anwendungsfall Übersicht



3.5.2 Anwendungsfall: Storyboard erstellen

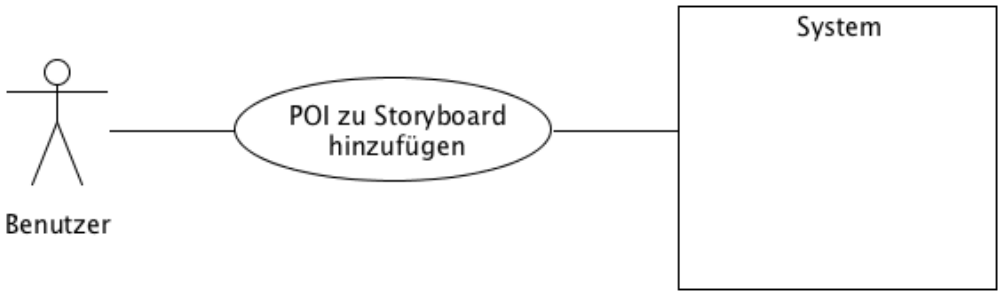
Abbildung 3.3: Anwendungsfall: Storyboard erstellen



Ziel	Ein neues Storyboard ist erstellt.
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich im Hauptmenü
Ablauf	• Der Benutzer wählt 'Neues Storyboard erstellen' aus dem Actionmenü • Das System zeigt dem Benutzer eine Liste der betehenden Objekte • Der Benutzer wählt ein bestehendes Objekt aus der Objektliste aus. • Das System fordert den Benutzer auf den Namen des Storyboards einzugeben. • Der Benutzer gibt den Namen für das neue Storyboard ein. • Das System erstellt das neue Storyboard unter dem vom Benutzer eingegebenen Namen
Nachbedingungen	Das neue Storyboard ist unter dem vom Benutzer eingegeben Namen erstellt worden.

3.5.3 Anwendungsfall: POI zu Storyboard hinzufügen

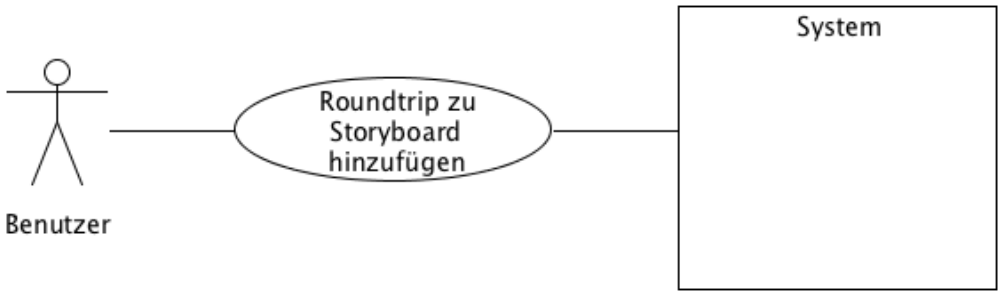
Abbildung 3.4: Anwendungsfall: POI zu Storyboard hinzufügen



Ziel	Der vom Benutzer ausgewählte POI wurde dem vom Benutzer gewählten Storyboard hinzugefügt
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich in der 'Storyboard anzeigen' Ansicht.
Ablauf	• Der Benutzer wählt 'POI hinzufügen' aus dem Actionmenü aus. • Das System zeigt dem Benutzer eine Liste der verfügbaren POIs an, die nach der Nähe zum Objekt des Storyboards sortiert ist. • Der Benutzer wählt einen POI aus der Liste aus. • Das System zeigt dem Benutzer die Medien des vom Benutzer ausgewählten POIs. • Der Benutzer selektiert alle Medien des gewählten POIs, die er in dem Storyboard anzeigen lassen möchte und bestätigt die Auswahl. • Das System fügt das gewählte POI mit der getroffenen Medienauswahl zum Storyboard hinzu.
Nachbedingungen	Dem aktuellen Storyboard wurde das gewählte POI mit selektierter Medien Konfiguration hinzugefügt

3.5.4 Anwendungsfall: Roundtrip zu Storyboard hinzufügen

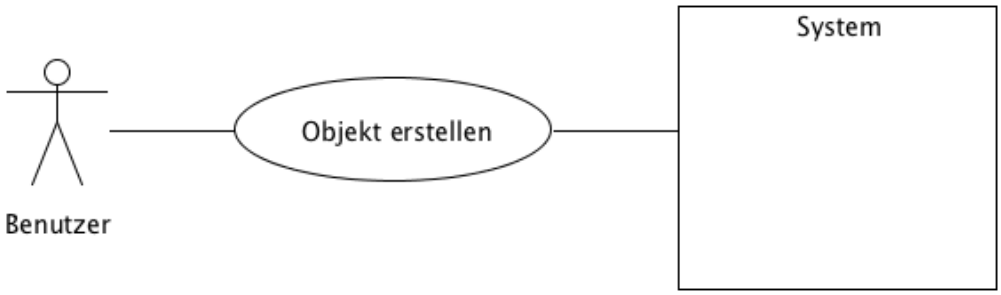
Abbildung 3.5: Anwendungsfall: Roundtrip zu Storyboard hinzufügen



Ziel	Der vom Benutzer ausgewählte Roundtrip wurde dem aktuellen Storyboard hinzugefügt
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich in der 'Storyboard anzeigen' Ansicht.
Ablauf	• Der Benutzer wählt 'Roundtrip hinzufügen' aus dem Actionmenü aus. • Das System zeigt dem Benutzer eine Liste der verfügbaren Roundtrips an, die nachder Nähe zum Objekt des Storyboards sortiert ist. • Der Benutzer wählt einen Roundtrip aus der Liste aus. • Das System fügt den gewählten Roundtrip zum Storyboard hinzu.
Nachbedingungen	Dem aktuellen Storyboard wurde der gewählte Roundtrip hinzugefügt.

3.5.5 Anwendungsfall: Objekt erstellen

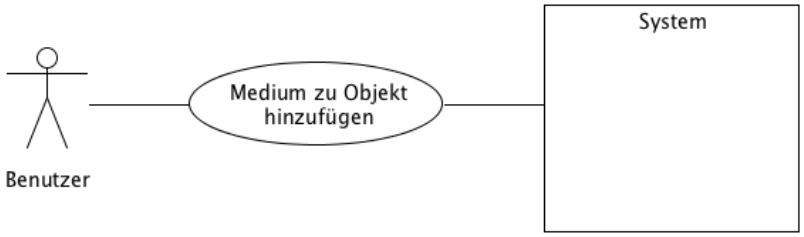
Abbildung 3.6: Anwendungsfall: Objekt erstellen



Ziel	Ein neues Storyboard ist erstellt.
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich im Hauptmenü
Ablauf	• Der Benutzer wählt 'Neues Objekt erstellen' aus dem Actionmenü • Das System versucht die aktuelle Adresse anhand der aktuellen GEO Informationen herauszufinden • Das System zeigt dem Benutzer die 'Neues Objekt erstellen' Ansicht an, in der die Adressinformationen durch den vorherigen Schritt gefüllt sind. • Der Benutzer gibt die Objektparameter des neuen Objekts ein und bestätigt die Eingaben. • Das System legt das neue Objekt mit den eingestellen Parametern und GEO Informationen an.
Nachbedingungen	Das neue Objekt ist erstellt.

3.5.6 Anwendungsfall: Medien zu Objekt hinzufügen

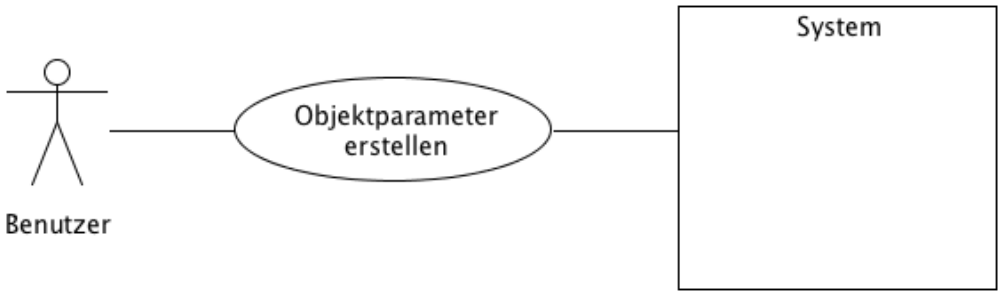
Abbildung 3.7: Anwendungsfall: Medien zu Objekt hinzufügen



Ziel	Dem aktuellen Objekt soll ein Medium hinzugefügt werden.
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich in der 'Objekt anzeigen' Ansicht.
Ablauf	• Der Benutzer wählt 'Medium hinzufügen' aus dem Actionmenü aus. • Das System zeigt dem Benutzer eine Liste der 'Neuen Medien' an. (Foto, Video, Audio) • Der Benutzer wählt eine Medienart aus der Liste. • Das System fragt den Benutzer ob er ein neues Medium der zuvor gewählten Art erstellen oder ein bestehendes dieser Art hinzufügen möchte. • Der Benutzer wählt eine der ihm gebotenen Möglichkeiten aus. • Das System lädt eine Auswahlansicht, wenn der Benutzer bestehende Medien hinzufügen möchte oder öffnet eine entsprechende Applikation um ein neues Medium der gewählten Art hinzuzufügen. • Der Benutzer selektiert oder erstellt ein Medium und bestätigt das Hinzufügen. • Das System fügt das neue Medium zum aktuellen Objekt hinzu.
Nachbedingungen	Das neue Medium wurde dem aktuellen Objekt hinzugefügt.

3.5.7 Anwendungsfall: Objektparameter erstellen

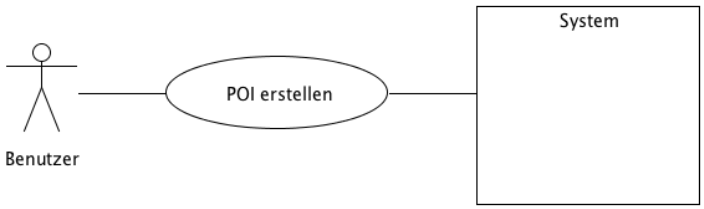
Abbildung 3.8: Anwendungsfall: Objektparameter erstellen



Ziel	Ein neues Storyboard ist erstellt.
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich im Hauptmenü oder in der Objektansicht
Ablauf	• Der Benutzer wählt 'Neuen Objektparameter erstellen' aus dem Actionmenü • Das System zeigt dem Benutzer die 'Neuer Objektparameter' Ansicht an. • Der Benutzer setzt den Namen und den Parametertypen für den neuen Parameter und beendet die Erstellung des neuen Objektparameters. • Das System erstellt den neuen Objektparameter.
Nachbedingungen	Ein neuer Objektparameter steht im System zur Verfügung.

3.5.8 Anwendungsfall: POI erstellen

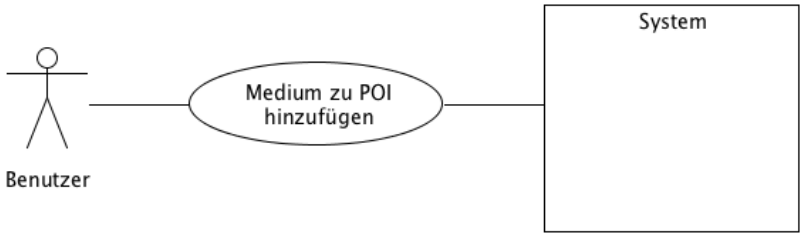
Abbildung 3.9: Anwendungsfall: POI erstellen



Ziel	Ein neuer POI wurde erstellt.
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich im Hauptmenü
Ablauf	• Der Benutzer wählt 'Neuen POI erstellen' aus dem Actionmenü • Das System fordert den Benutzer auf, zu entscheiden, ob er einen POI durch Locationbased Services suchen und hinzufügen möchte oder ob er einen eigenen erstellen möchte. • Der Benutzer wählt eine der Beiden Möglichkeiten aus und fährt fort. • Hat der Benutzer 'Eigenen POI erstellen' ausgewählt, zeigt das System dem Benutzer ein Formular an, in dem dieser einen Namen für das neue POI eingibt und es Geo Tagged. Hat der Benutzer 'Locationbased Services benutzen' gewählt, zeigt das System ihm eine Liste von sich in der Nähe befindlichen POIs aus dem Locationbased Service an. • Der Benutzer nimmt seine Auswahl/Konfiguration vor und bestätigt seine Auswahl/Konfiguration. • Das System fügt den neuen POI mit entsprechender Konfiguration hinzu.
Nachbedingungen	Ein neues POI wurde dem System hinzugefügt.

3.5.9 Anwendungsfall: Medien zu POI hinzufügen

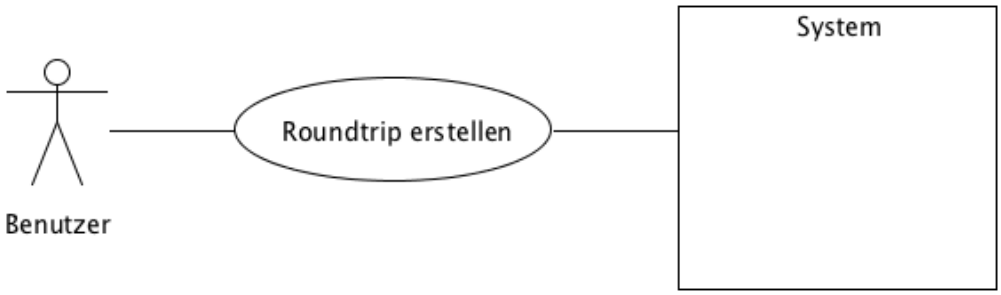
Abbildung 3.10: Anwendungsfall: Medien zu POI hinzufügen



Ziel	Dem aktuellen POI soll ein Medium hinzugefügt werden.
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich in der ’POI anzeigen’ Ansicht.
Ablauf	• Der Benutzer wählt ’Medium hinzufügen’ aus dem Actionmenü aus. • Das System zeigt dem Benutzer eine Liste der ’Neuen Medien’ an. (Foto, Video, Audio) • Der Benutzer wählt eine Medienart aus der Liste. • Das System fragt den Benutzer ob er ein neues Medium der zuvor gewählten Art erstellen oder ein bestehendes dieser Art hinzufügen möchte. • Der Benutzer wählt eine der ihm gebotenen Möglichkeiten aus. • Das System lädt eine Auswahlansicht, wenn der Benutzer bestehende Medien hinzufügen möchte oder öffnet eine entsprechende Applikation um ein neues Medium der gewählten Art hinzuzufügen. • Der Benutzer selektiert oder erstellt ein Medium und bestätigt das Hinzufügen. • Das System fügt das neue Medium zum aktuellen POI hinzu.
Nachbedingungen	Das neue Medium wurde dem aktuellen POI hinzugefügt.

3.5.10 Anwendungsfall: Roundtrip erstellen

Abbildung 3.11: Anwendungsfall: Roundtrip erstellen



Ziel	Ein neuer Roundtrip soll erstellt werden.
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich im Hauptmenü
Ablauf	• Der Benutzer wählt 'Neuen Roundtrip erstellen' aus dem Actionmenü • Das System zeigt dem Benutzer eine Maske, in der er einen Namen für den neuen Roundtrip eingeben kann. • Der Benutzer gibt einen Namen für den neuen Roundtrip ein und bestätigt diese Eingabe. • Das System erstellt einen neuen Roundtrip unter dem vom Benutzer eingegebenen Namen.
Nachbedingungen	Der neue Roundtrip wurde erstellt.

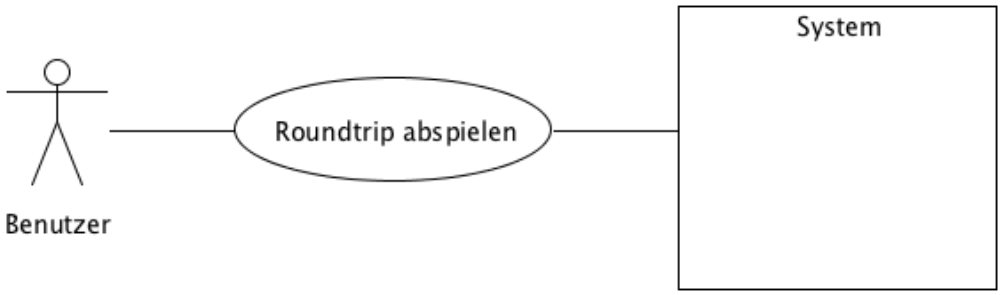
3.5.11 Anwendungsfall: POI zu Roundtrip hinzufügen

Abbildung 3.12: Anwendungsfall: POI zu Roundtrip hinzufügen

Ziel	Dem aktuellen Roundtrip soll ein POI hinzugefügt werden.
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich in der 'Roundtrip anzeigen' Ansicht.
Ablauf	• Der Benutzer wählt 'POI hinzufügen' aus dem Actionmenü aus. • Das System zeigt dem Benutzer eine Liste der verfügbaren POIs • Der Benutzer wählt einen POI aus der Liste aus. • Das System zeigt dem Benutzer die Medien des vom Benutzer ausgewählten POIs. • Der Benutzer selektiert alle Medien des gewählten POIs, die er in dem Roundtrip anzeigen lassen möchte und bestätigt die Auswahl. • Das System fügt den gewählten POI mit der getroffenen Medienauswahl zum Roundtrip hinzu.
Nachbedingungen	Dem aktuellen Roundtrip wurde der gewählte POI mit selektierter Medien Konfiguration hinzugefügt

3.5.12 Anwendungsfall: Roundtrip abspielen

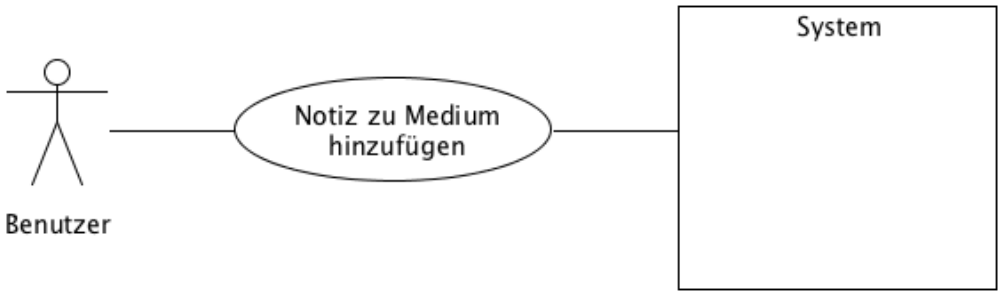
Abbildung 3.13: Anwendungsfall: Roundtrip abspielen



Ziel	Der aktuelle Roundtrip soll abgespielt werden
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich in der 'Roundtrip anzeigen' Ansicht.
Ablauf	• Der Benutzer wählt 'Roundtrip abspielen' aus dem Actionmenü aus. • Das System spielt den Roundtrip ab
Nachbedingungen	Der ausgewählte Roundtrip wurde abgespielt.

3.5.13 Anwendungsfall: Notiz zu Medium hinzufügen

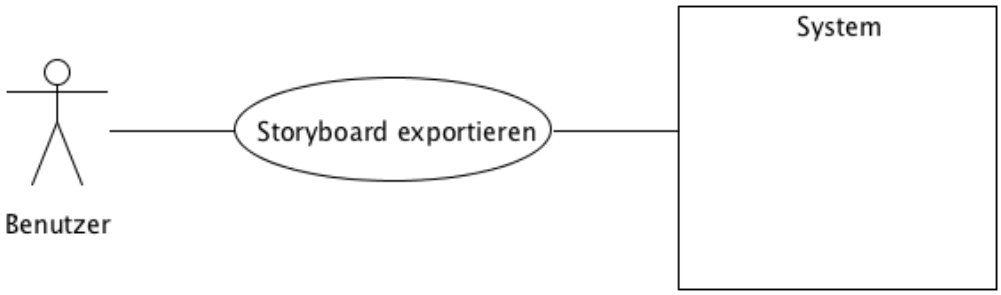
Abbildung 3.14: Anwendungsfall: Notiz zu Medium hinzufügen



Ziel	Zu dem aktuell gewähltem Medium soll eine Notiz hinzugefügt werden.
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich in der 'Medium anzeigen' Ansicht
Ablauf	• Der Benutzer wählt 'Notiz hinzufügen' aus dem Actionmenü • Das System zeigt dem Benutzer eine Maske, in der er eine Notiz eingeben kann. • Der Benutzer gibt eine Notiz zu dem aktuellen Medium ein und bestätigt diese. • Das System speichert zu dem aktuell gewähltem Medium die zuvor eingegebene Notiz.
Nachbedingungen	Dem aktuell gewähltem Medium wurde eine Notiz hinzugefügt.

3.5.14 Anwendungsfall: Storyboard exportieren

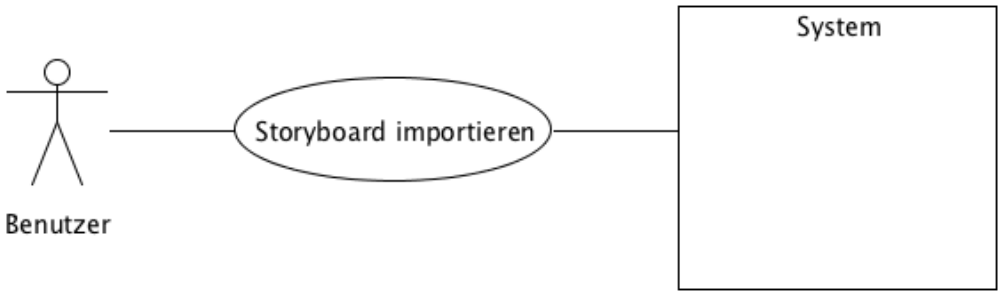
Abbildung 3.15: Anwendungsfall: Storyboard exportieren



Ziel	Das aktuell gewählte Storyboard soll exportiert werden.
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich in der 'Storyboard anzeigen' Ansicht und mindestens ein Exportprovider wurde konfiguriert.
Ablauf	• Der Benutzer wählt 'Exportieren' aus dem Actionmenü • Das System zeigt dem Benutzer eine Liste aller vom Benutzer konfigurierten Exportmöglichkeiten / der Exportprovider • Der Benutzer wählt einen dieser Provider aus und bestätigt seine Auswahl. • Das System startet den gewählten Provider, welcher den Benutzer falls nötig, durch weitere Schritte führt. • Der Benutzer führt den Export im Exportprovider durch. • Das System hat das gewählte Storyboard durch den gewählten Provider exportiert.
Nachbedingungen	Das aktuell gewählte Storyboard wurde exportiert.

3.5.15 Anwendungsfall: Storyboard importieren

Abbildung 3.16: Anwendungsfall: Storyboard importieren



Ziel	Ein Storyboard soll importiert werden.
Akteur	Benutzer, System
Vorbedingungen	Der Benutzer befindet sich im Hauptmenü der Applikation und mindestens ein Importprovider wurde konfiguriert
Ablauf	• Der Benutzer wählt 'Importieren' aus dem Actionmenü • Das System zeigt dem Benutzer eine Liste aller vom Benutzer konfigurierten Importmöglichkeiten / Importprovidern • Der Benutzer wählt einen dieser Provider aus und bestätigt seine Auswahl. • Das System startet den gewählten Provider, welcher den Benutzer falls nötig, durch weitere Schritte führt. • Der Benutzer führt den Import im Importprovider durch. • Das System hat ein durch den gewählten Provider gewähltes Storyboard importiert.
Nachbedingungen	Ein Storyboard wurde importiert.

3.6 Abgrenzung

Da es sich, wie im Kapitel 1.1 beschrieben um eine Proof of Concept Applikation handelt, ist eine Integration in eine bestehende Infrastruktur nicht vorgesehen. Des Weiteren wird aufgrund der Komplexität auf die Import und Export Funktion verzichtet. Die Anbindung an die LocationBasedServices wird in dieser Arbeit exemplarisch an Google Places durchgeführt, wobei eine mögliche Erweiterung in der Konzeption berücksichtigt werden soll.

3.7 Zusammenfassung

Durch die in diesem Kapitel durchgeführte Analyse des Ist-Zustands konnten Akteure und Entitäten herausgefiltert werden. Aus den definierten Zielen wurden Anwendungsfälle generiert, die den Funktionskatalog des MobileStoryboards definieren. Im Weiteren wurde der Zweck von Locationbased Services erklärt. In den Anwendungsfällen wurde definiert wie der Benutzer durch Interaktion mit der Applikation, Storyboards und deren Bestandteile erstellen kann. Aus den Workflows der Anwendungsfälle wurden Vorgaben an die GUI abgeleitet.

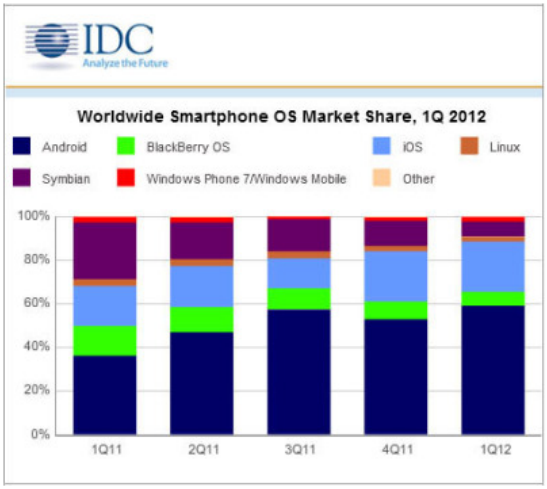
4 Design

In diesem Kapitel werden neben den Anforderungen an Soft- Und Hardwareausstattung auch die verwendeten Design Patterns, sowie die für diese Arbeit wichtigsten Android Komponenten beschrieben. Weiterhin wird der Komponentenschnitt der Applikation betrachtet, wobei die Aufgaben der einzelnen Komponenten und ihr Zusammenspiel erklärt werden sollen.

4.1 Anforderung an Soft- und Hardware Ausstattung

Unter Berücksichtigung der, in den bisherigen Kapiteln definierten, Anforderungen, lässt sich ein Katalog von Eigenschaften formulieren, der die Vorgaben an die Zielplattform bestimmt. Da die an die Hardware gerichteten Anforderungen wie Mobilität, Kamerafunktion, Positionsbestimmung sowie Internetfähigkeit durch alle modernen Smartphone Typen gegeben sind, kommen grundsätzlich alle modernen Smartphones für die Implementierung in Frage. Ein

Abbildung 4.1: IDC: Worldwide Smartphone OS Market Share Q1 2012



großer Unterschied bietet sich hier durch das zugrundeliegende Betriebssystem. Neben Apples

iPhone bieten sich hier noch Android Smartphones, BlackBerry Geräte sowie Windows Phones an. Wichtig sollte allerdings sein, neben einer weiten Verbreitung auch auf eine offene Softwarelösung zu bestehen. Dies bietet eine Vielfalt von unterschiedlichen Geräten, die für jederman zugänglich und erschwinglich sind. Als einziges offenes Betriebssystem bietet das Android Betriebssystem außerdem alle benötigten Schnittstellen, um die definierten Anforderungen an die Applikation umsetzen zu können. Ein Marktanteil von mehr als 55 Prozent (siehe Grafik von IDC) bietet eine mehr als hinreichende Verbreitung. Weiterhin bietet Android durch seine Interkomoponenten Kommunikation auch noch die Möglichkeit, eine Applikation nahtlos in das bestehende System zu integrieren und mit den installierten Applikation kommunizieren zu lassen. Der Vorteil hierbei ist, dass eine Applikation nicht wissen muss, welche anderen Applikationen installiert sind, um eine bestimmte Aufgabe zu erledigen, sowenig wie man einen bestimmten Teil (wie eine Kamera) neu implementieren muss. Es reicht zu wissen, was man auf dem System tun möchte und das System führt die entsprechende Aktion aus. Möchte ein Benutzer beispielsweise eine speziellere Kameraapplikation installieren, müssen die Applikationen die die Kamera benutzen nicht neu implementiert werden, sondern der Android Kern lässt den Benutzer zwischen der alten und der neuen Kameraapplikation wählen. Das offene Betriebssystem, die weite Verbreitung, sowie die hervorragende Kommunikation zwischen den Komponenten waren die Gründe, sich für eine Entwicklung auf dem Android Betriebssystem zu entscheiden.

4.2 Programmiermodell: Android

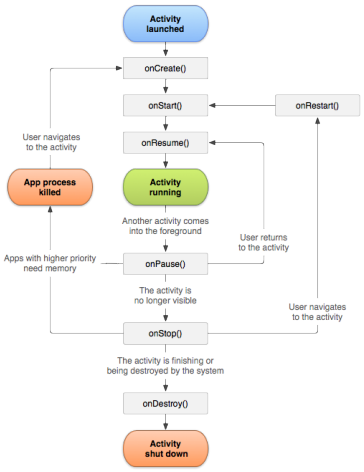
Eine Android Applikation besteht aus unterschiedlichen Komponenten die lose miteinander gekoppelt sind. Die Kommunikation untereinander passiert asynchron und kann mit einer Blackboard Architektur verglichen werden. In dieser können die Komponenten Dienste oder Aktionen anbieten und anfordern. Der Android Kern selbst verwaltet das Matching der Anforderungen auf Angebote und startet die gewünschte Komponente, die der Anforderung entspricht. Im Folgenden werden die Komponenten, die in dieser Arbeit verwendet werden, erläutert und es wird ein Bezug zu dieser Arbeit hergestellt.

4.2.1 Activities

Activities verbinden die GUI Elemente mit dem Programmcode. Sie sind das, was der Benutzer sieht und mit dem er agieren kann. In einer Applikation gibt es meist mehrere Activities, die für einzelne Aufgaben wie das Anzeigen oder das Bearbeiten von Daten zuständig sind.

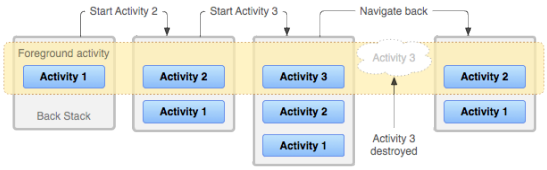
Aus einer Activity heraus können andere Activities gestartet werden. Die Kontrolle des Lebenszyklus einer Activity, die in der folgenden Abbildung gezeigt wird, obliegt dem Android Kern, der auch das Starten und Beenden einer Activity übernimmt.

Abbildung 4.2: Android: Activity Lifecycle



Startet eine Activity A eine Activity B, so wird die Activity A gestoppt und die Activity B gestartet. Hierbei behält das System die für das Starten von B verantwortliche Activity auf dem so genannten Back-Stack vor. Wird Activity B verlassen, startet das System die Activity, die als letztes auf den Back-Stack gelegt wurde (A). Die Activities sind untereinander lose gekoppelt. Wenn sie eine andere Activity starten möchten, teilen sie dies dem System mit, welches den gesamten Startprozess übernimmt. Dieser Lebenszyklus einer Activity, wird in obiger Abbildung noch einmal veranschaulicht.

Abbildung 4.3: Android: Backstack



4.2.2 Fragments

Wenn man eine Activity baut und einen oder mehrere Teile aus dieser in anderen Activities wiederverwenden möchte, kann man sich der Fragments bedienen. Sie sind Teile des Benutzerinterfaces, die ein eigenes Design sowie einen eigenen Lebenszyklus besitzen und eigenständig auf Events, wie das Klicken auf ein Element, reagieren können. Ihr Lebenszyklus ist gebunden an den Lebenszyklus der Activity, welche sie beinhaltet. Wird die Host Activity zerstört so werden auch alle Fragments zerstört, die sich in der Activity befinden. Außerdem kann eine Activity Fragments zur Laufzeit hinzufügen oder entfernen.

In dieser Arbeit werden Fragments benutzt um die Auflistung der unterschiedlichen Entitäten im Hauptmenü als Tabs zu realisieren. Fragments bieten sich hier an, da das horizontale Scrollen durch die unterschiedlichen Tabs, mit Androidhilfsmitteln möglich ist.

4.2.3 Services

Services dienen dazu, Zeitintensive Aktionen zu übernehmen, die im Hintergrund laufen sollen und keine GUI benötigen. Sie laufen auch dann im Hintergrund, wenn die Applikation, welche den Service gestartet hat nicht mehr aktiv ist. Das Starten eines Services kann manuell oder über eine Bindung an eine Komponente geschehen. So eine Bindung kann durch mehrere Komponenten aus unterschiedlichen Applikationen erfolgen.

4.2.4 Intents

Die Interapplikation- sowie die Interkomponenten- kommunikation in Android erfolgt über Intents. Intents sind Vorhaben die genutzt werden um eine bestimmte Aktion durchzuführen. Es ist möglich dem Ziel einer Aktion Daten über ein Intent zu übermitteln. Diese Daten heißen Extras. Um eine andere Applikation zu starten, verwendet eine Activity ein Intent und reichert dieses mit Informationen über das Vorhaben an. Der Android Kern empfängt das angereicherte Intent und startet die Applikation, welche die Aktion umsetzen kann. Ist auf dem System mehr als eine Applikation fähig die Aktion durchzuführen, kann der Benutzer eine der Applikationen wählen, und das System startet diese. Anders als bei dem eben beschriebenen implizierten Starten von Actvities, können Activities auch direkt addressiert werden. Bei diesem expliziten Starten, wird dem Intent der Klassenname der zu startenden Activity mitgegeben. Der Android Kern startet dann die entsprechende Activity.

Die Mobile Storyboard Applikation benutzt Intents nicht nur, um innerhalb der Applikation durch die unterschiedlichen Activities zu navigieren, sondern auch um Bilder auszuwählen oder die Kamera App zu starten um Bilder aufzunehmen.

4.2.5 Intent-Filter

Um eine Applikation auf Aktionen reagieren zu lassen, kann man Intent-Filter einrichten. Diese Filter hören auf Intents und werden für Activities oder Services hinterlegt. Sie müssen für mindestens eine Aktion eingerichtet werden und können durch Kategorien und Datenfilter weiter parametrisiert werden. In einem Datenfilter können der MIME-Type sowie das Schema/Protokoll der über das Intent übermittelten Daten limitiert werden. Über Kategorien kann festgelegt werden von welchem Typ die Komponente ist, welche den Intent-Filter benutzt. Eine Komponente die einen Inten-Filter mit der Kategorie CATEGORY_BROWSABLE einsetzt, besagt, das diese Komponente in der Lage ist, Daten anhand eines Links darzustellen. Eine Applikation die folgenden Intent-Filter in einer der Definitionen seiner Komponenten benutzt, reagiert damit auf das Öffnen von Bildern, die über einen link über das HTTP Protokoll angezeigt werden sollen.

```
<intent-filter>
        <action android:name='android.intent.action.VIEW' android:scheme='http' />
        <category android:name='android.intent.category.DEFAULT' />
        <category android:name='android.intent.category.BROWSABLE' />
        <data android:mimeType='image/*' />
</intent-filter>
```

4.2.6 Permissions

Jede Applikation läuft in einer Sandbox, in der Sie vor Zugriff durch andere Applikationen abgeschottet ist. Sie hat eine eindeutige Benutzer sowie Gruppen ID, unter der der Prozess der Applikation ausgeführt wird. Die Sandbox der Applikation hat von Hause aus, weder die Berechtigung auf das Internet zuzugreifen, noch Hardwarefeatures zu benutzen oder zum Beispiel SMS zu empfangen. Um einer Applikation und ihrer Sandbox mehr Berechtigungen und Features zu erlauben, muss man statisch Permissions vergeben, über die das System dann weiß, was die Applikation darf. Bei der Installation über den Android Market, muss der Benutzer nun erst erlauben, dass die Applikation diese Berechtigung benutzen darf. Ohne dies, kann er die Applikation nicht installieren. Dies gilt nicht für Testgeräte, da die Applikation hier nicht über den Market verteilt wird. Permissions werden in dieser Arbeit verwendet um der Applikation zum Beispiel die Benutzung des Internets oder das Erhalten der aktuellen Position über GPS zu erlauben.

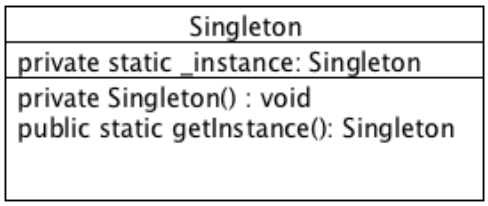
4.3 Patterns

Wie in der Software Entwicklung hinreichend bekannt, kann die Verwendung von Design Patterns dazu dienen, Applikationen wartbar und erweiterbar zu gestalten. Designpatterns dienen unter anderem dazu, nicht ständig Innovationen erfinden zu müssen und dabei weit verbreitete Fehlerquellen zu erliegen. Sie enthalten Lösungen zu entsprechenden Problemen in der objektorientierten Programmierung und können weiterhin das Lesen und Verstehen von Programmcode bei Erweiterungsarbeiten erleichtern. Die folgenden Designpatterns fanden Einzug in die Konzeption der Arbeit und werden im Folgenden mit entsprechendem Bezug zur Arbeit beschrieben.

4.3.1 Singleton

Von Klassen die das Singleton Pattern implementieren, kann es zur Laufzeit nur eine Instanz geben. Die Initialisierung dieser Klassen kann über Eager oder Lazy initialization passieren. Bei der Eager initialization, passiert die Instanziierung der Klasse beim Laden der Klasse. Bei der Lazy initialization erfolgt die Instanziierung erst beim Aufruf einer statischen Methode. Das

Abbildung 4.4: Singleton Pattern

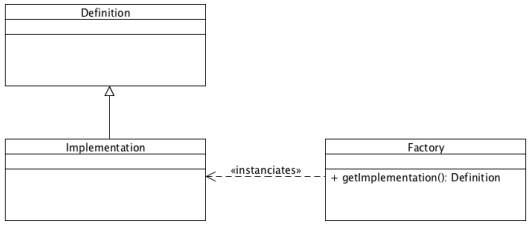


Singleton Pattern wird in dieser Arbeit bei der Storyboard Komponente angewendet. Durch das Implementieren dieses Patterns erhält man in der gesamten Applikation Zugriff auf die anderen Komponenten, die sie bei ihrer eigenen Instanziierung instanziert und gegebenfalls konfiguriert.

4.3.2 Factory

Beim Factorypattern werden über eine separierte Klasse (Die Factory Klasse) konkrete Implementationen einer abstrakten Klasse oder eines Interfaces instanziiert. Es ist somit nicht nötig,

Abbildung 4.5: Factory Pattern

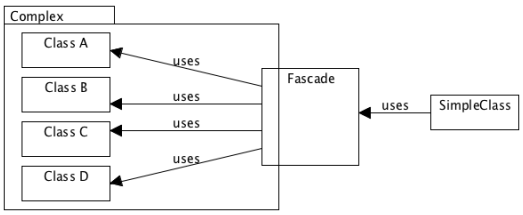


die konkrete Instanziierung an der Stelle vorzunehmen, wo die Instanz benötigt wird, so wenig wie es nach außen nötig ist zu wissen, wie die konkrete Instanziierung vonstatten geht. Das Mobile Storyboard benutzt das Factorypattern in der LBS Komponente (siehe Kapitel 4.4.3). Es wird benutzt um die verschiedenen LocationBasedService Provider wartbar und erweiterbar zu halten. Aus denselben Gründen wird das Factorypattern auch bei der Export Komponente angewandt (siehe Kapitel 4.4.5). Hierdurch lässt sich die Applikation durch wenig Aufwand durch neue Exporter erweitern.

4.3.3 Fassade

Durch eine Fassade ist es möglich, eine Abstraktionsschicht zwischen einem Subsystem und den Klassen einzurichten, die das Subsystem aufrufen. Dies macht Sinn, wenn es sich um ein sehr komplexes oder technisches Subsystem handelt. Die Fassade bietet dann eine Schnittstelle an und deligiert die Aufrufe entsprechend an eine oder mehrere Klassen innerhalb des Subsystems. Um die relativ komplexe Datenbankkomponente nach außen einfacher zu

Abbildung 4.6: Fassade Pattern



gestalten, implementiert diese Komponente das Fassade Pattern. Sie implementiert generische

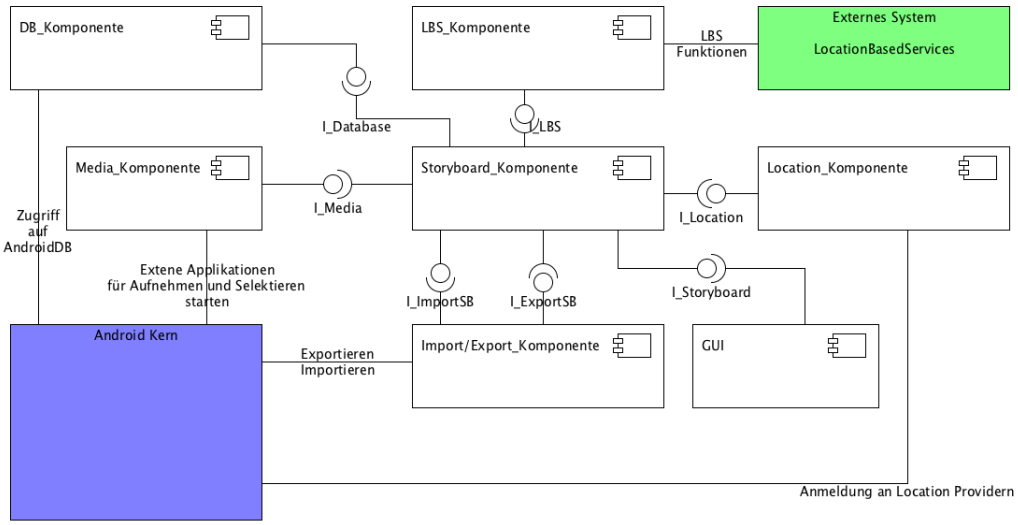
Methoden und deligiert die Aufrufe anhand der übergebenen Klassen an die entsprechenden Klassen innerhalb der Komponente weiter. (siehe Kapitel ??).

4.4 Komponentenschnitt

Die MobileStoryboard Applikation besteht aus Komponenten die in unterschiedlicher Intensität mit Android Komponenten zusammenarbeiten müssen, um ihre Aufgaben zu erfüllen.

Um dies zu bewerkstelligen, hält jede Komponente den Applikationskontext als Referenz. Dies ermöglicht es auch außerhalb einer Android Komponente, auf Funktionen wie Datenbank Management, etc. zugreifen zu können.

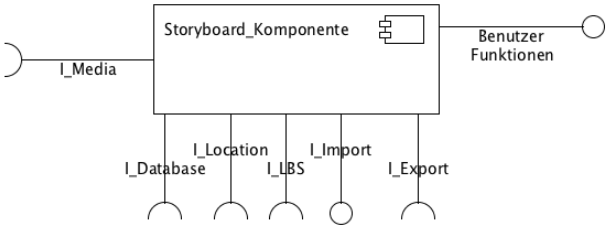
Abbildung 4.7: Komponenteschnitt



Die einzelnen Aufgaben und Funktionen der Komponenten werden im Folgenden erläutert.

4.4.1 Storyboard Komponente

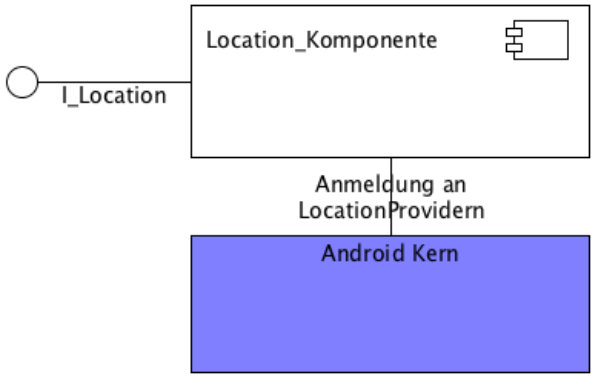
Abbildung 4.8: Komponente: Storyboard



Aufgaben	Die Storyboardkomponente dient als Verbindungskomponente. Über das I_Storyboard Interface erhalten Android Komponenten Zugriff auf die in der Storyboard Komponente gekoppelten anderen Komponenten. Die Kopplung an die anderen Komponenten geschieht nach der Instanziierung in der Konfiguration dieser Komponente.
Schnittstellen	I_Media, I_Location, I_LBS, I_Import, I_Export, I_Database
Patterns	Singleton

4.4.2 Location Komponente

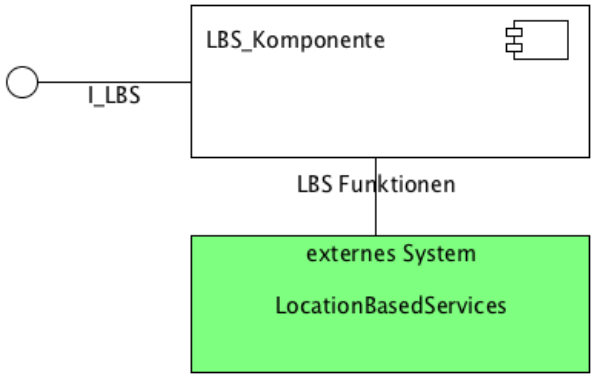
Abbildung 4.9: Komponente: Location



Aufgaben	Um die aktuelle Position des Gerätes heraus zu finden, bedient sich diese Komponente verschiedener Hilfsmittel aus dem Android Framework, um bei einem Positionswechsel über eine solche informiert zu werden. Diese Komponente meldet sich bei dem System für alle vorhanden Provider an, die dieser Komponente die aktuelle Position mitteilen können. Somit bekommt man auch eine ungefähre Position, wenn man gerade kein GPS zur Verfügung hat. Das Aktualisieren der aktuellen Position passiert asynchron, per Notification. Der Abruf der aktuellen Position über diese Komponente passiert synchron.
Schnittstellen	I_Location

4.4.3 LocationBasedService Komponente

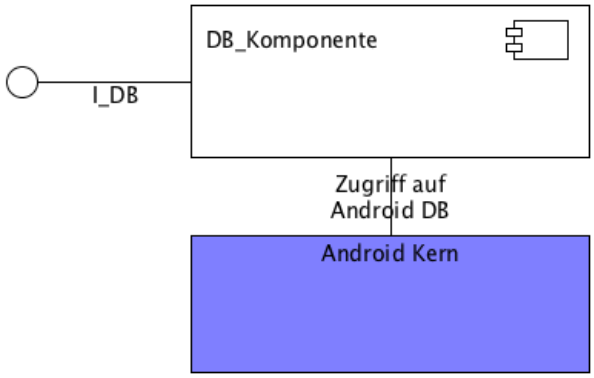
Abbildung 4.10: Komponente: LocationBasedService



Aufgaben	Über diese Komponente erhält man Zugriff auf die konkreten Implementierungen der unterschiedlichen verfügbaren Location Based Services. Jede Implementierung verfügt durch ihr gemeinsames Interface I_Service über Methoden, um den konkreten Service nach POIs zu durchsuchen oder über Details zu einem POI zu befragen. Die Suche nach POIs erfolgt über einen Suchstring und eine in der Applikation eingestellte Genauigkeit. Jeder POI in der Ergebnisliste hat eine, für den LBS, eindeutige Kennung. Diese Kennung dient als Parameter für die Detailanfrage, welche als Ergebnis Zusatzinformationen wie Bewertungen, Kommentare oder Ähnliches enthalten kann.
Schnittstellen	I_LBS I_Service
Patterns	Factory

4.4.4 Datenbank Komponente

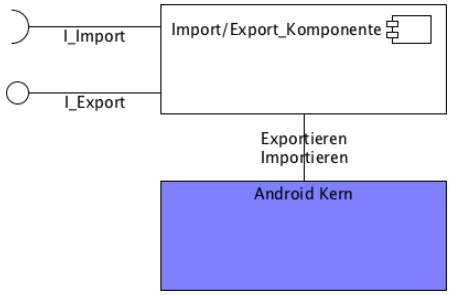
Abbildung 4.11: Komponente: Datenbank



Aufgaben	Diese Komponente beinhaltet alle Klassen, die in Zusammenarbeit mit dem Android System die Verwaltung der Datenbank und ihrer Daten übernehmen. Sie hat ein festes Repertoire an Methoden, die sie als Fassade an die entsprechenden Klassen deligiert. Weiterhin abstrahiert sie nach Außen von echten Datenbankabfragen und übernimmt die Generierung von Datensatz repräsentierenden Objekten.
Schnittstellen	I_Database
Patterns	Fassade

4.4.5 Import/Export Komponente

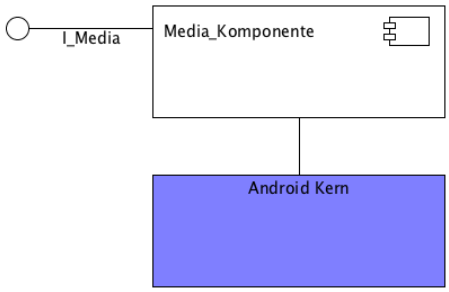
Abbildung 4.12: Komponente: Import/Export



Aufgaben	Diese Komponente ist zuständig für das Importieren und Exportieren eines Storyboards. Im Falle eines Exports serialisiert sie ein Storyboard und alle Bestandteile, wie das verknüpfte Objekt, Roundtrips und Medien und generiert aus diesen Daten ein ZIP Paket. Durch unterschiedliche Exporter kann dieses Paket nun zum Beispiel als Anhang per E-Mail verschickt oder auf einen zuvor konfigurierten FTP-Server hochgeladen werden. Die unterschiedlichen Exporter werden hier durch das Factorypattern instanziiert. Als Importer ist die Komponente im Android System per Intent-Filter für das Öffnen von ZIP-Dateien registriert. Wird die Applikation nun aus einer anderen gestartet, entpackt diese Komponente die mitgegebene Datei und deserialisiert diese. Die deserialisierten Daten werden nun in der Applikation registriert und der Benutzer kann auf die Daten zugreifen, als hätte er dieses Storyboard selber generiert.
Schnittstellen	I_ImportSB, I_ExportSB
Patterns	Factory

4.4.6 Media Komponente

Abbildung 4.13: Komponente: Media



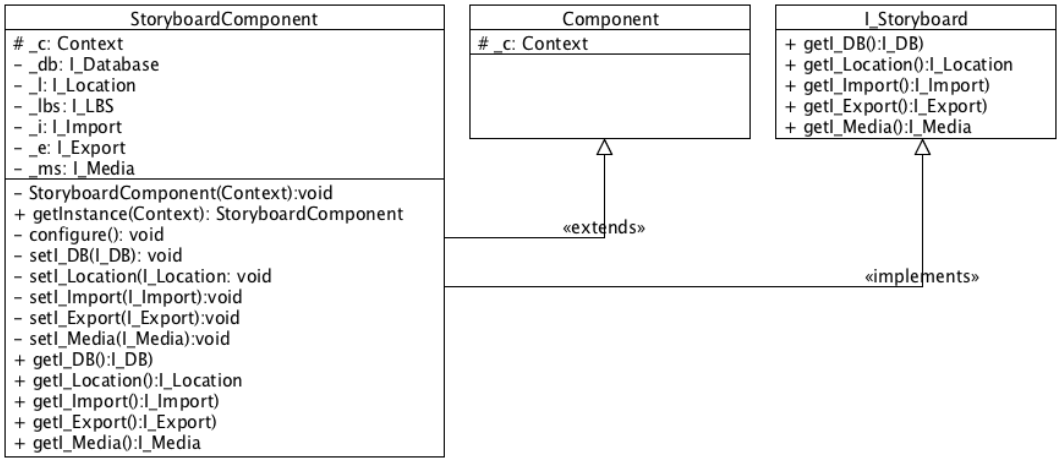
Aufgaben	Um Medien zu Objekten oder Storyboards hinzuzufügen, muss es eine Möglichkeit geben Medien zu erstellen oder bestehende Medien zu importieren. In Android ist es möglich Applikationen zu einem bestimmten Vorhaben zu starten, ohne zu wissen welche Applikationen das gewünschte Vorhaben unterstützen. Diese Möglichkeit nutzt die Medien Komponente und übernimmt den Koordinator-part zwischen der Applikation und dem Android Framework. Soll zum Beispiel ein Bild zu einem Objekt hinzugefügt werden, entscheidet sich der Benutzer für das Aufnehmen oder das Importieren eines Mediums. Die Mediakomponente listet dem Benutzer dann die möglichen Applikationen auf, die sein Vorhaben in die Tat umsetzen können. Wurde das Vorhaben dann, in der externen Applikation, umgesetzt übernimmt die Mediakomponente das Umspeichern/Kopieren der Medien und gibt die interne ID zurück. Diese kann dann von den anderen Komponenten weiterverwendet werden.
Schnittstellen	I_Media

4.5 Klassenmodelle

4.5.1 Storyboardkomponente

Die Storyboard Komponente implementiert das Singleton Pattern und wird über die Methode getInstance instanziiert. Wie in der Abbildung zu sehen, benötigt diese Methode einen Kontext. Dieser Kontext ist der Applikationkontext, welcher benötigt wird um die in dieser Applikation verwendeten Ressourcen, wie die verwendete SQLite Datenbank, zu gelangen. Wie die Sto-

Abbildung 4.14: Klassendiagramm der Storyboard Komponente

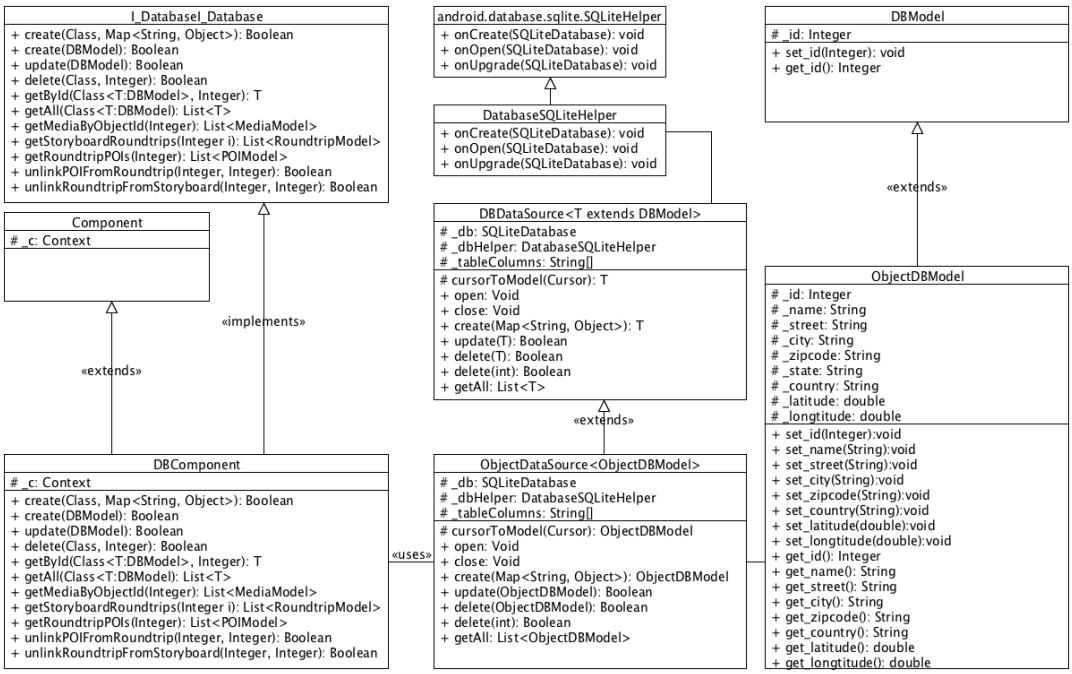


ryboardkomponente, spezialisieren auch alle anderen Komponenten der Mobile Storyboard Applikation die Component Klasse und benötigen den Applikationskontext. Beim Initialisieren der Storyboardkomponente instanziiert sie alle anderen Komponenten und übergibt den Applikationskontext. Es ist nun möglich, an jeder Stelle der Applikation an die Referenzen der gekoppelten Komponenten zu gelangen.

4.5.2 Datenbankkomponente

Über die Datenbankkomponente wird die Persistenz der Storyboards (siehe Kapitel 3.3) und der mit ihnen verknüpften Entitäten sichergestellt. Die Klasse DBComponent dient als Fassade dieser Komponente. Sie beinhaltet generische Methoden für das Manipulieren von Datensätzen über repräsentierende Objekte und deligiert diese Anhand der Klasse des übergebenen Objektes an die entsprechende DataSource-Klasse. In den DataSource-Klassen werden die echten

Abbildung 4.15: Klassendiagramm der Datenbankkomponente



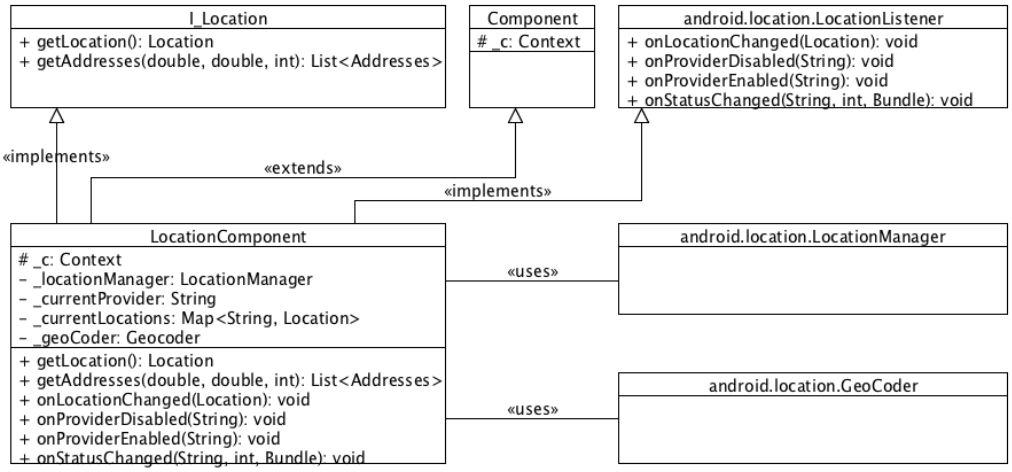
Die Grafik wurde aufgrund der Komplexität der Datenbank Komponente auf die Objekt Entität beschränkt. Die Delegation der Methodenaufrufe durch die DBComponent Klasse für die restlichen Entitäten unterscheidet sich ausschließlich durch die übergebenen Klassen.

Datenbankabfragen an die SQLite Datenbank geschickt. Die Übersetzung von einem Datensatz zu einem repräsentierenden Objekt geschieht hier ebenfalls. Die DatabaseSQLiteHelper Klasse, die der Komponente als Referenz vorliegt, wird zum Öffnen und Schließen der Datenbank benutzt. Sie übernimmt außerdem das Erstellen der Datenbank sowie deren Tabellen.

4.5.3 Location Komponente

Die Location Komponente benutzt unterschiedliche Funktionen aus Android, um den aktuellen Standort herauszufinden und diesen in eine Adresse zu kodieren. Zunächst implementiert sie das Interface LocationListener und ist somit dazu bereit auf Positionsänderungen zu reagieren. Um über solche Änderungen informiert zu werden, meldet sie sich über den Location Manager

Abbildung 4.16: Klassendiagramm der Location Komponente



bei allen auf dem Gerät vorhanden LocationProvidern an, die der Komponente dann ihre Positions- oder Statusänderung mitteilen. Sobald eine solche Änderung durch die Komponente wahrgenommen wurde, merkt sie sich diese, mit einem Vermerk über den entsprechenden Provider. Als Quelle bzw. Provider der aktuellen Position kann einer der folgenden 3 angegeben sein. Der GPS-Provider nutzt Sateliten um eine Ortsbestimmung vorzunehmen. Der Networkprovider ermittelt den aktuellen Standort über die aktuelle Telefonzellen Situation und/oder über WirelessLAN's in Reichweite. Als letztes ist hier der Passiveprovider zu nennen, welcher nicht aktiv nach Positionen sucht, sondern sich der anderen Provider bedient. Dies bedeutet, dass wenn eine andere Applikation eine Ortsbestimmung über den Networkprovider ausgeführt hat, würde man diese Position über den Passiveprovider erhalten können. Beim Abfragen der Location über die Location Komponente gibt diese die beste gefundene Location zurück. Wobei gilt:

Der GPS Provider ist genauer als der Network Provider.
Der Network Provider ist genauer als der Passive Provider.

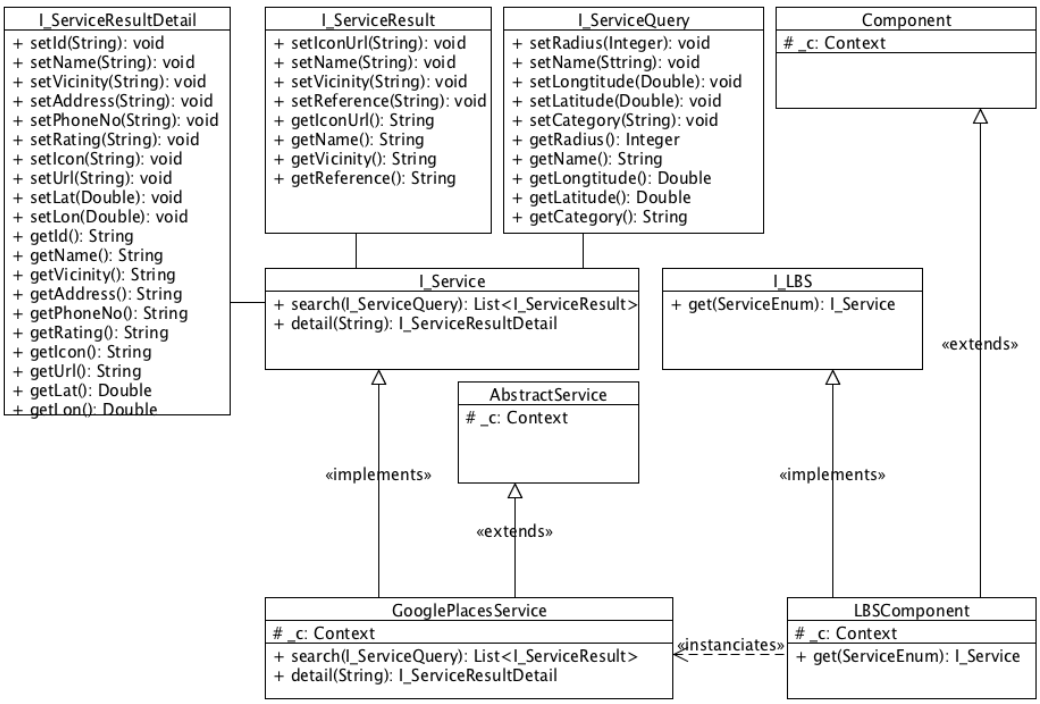
Der Passive Provider ist genauer als Nichts.

Die Komponente nutzt den GeoCoder, um eine Position - angegeben durch Latitude und Longtitude Werte - in eine Adresse umzuwandeln. Die Ergebnisliste kann auf eine fest eingestellte Anzahl von Adressen limitiert werden.

4.5.4 LBS Komponente

Die LBS Komponente bietet, als Implementierung des Factorypatterns, nur die getService Methode nach außen an. Indem man diese Methode mit einem ServiceEnum aufruft, erhält man eine Implementierung des I_Service Interfaces zurück, den man dann nach POIs durchsuchen kann. Die Zuordnung von einem Service Enum zu der entsprechenden Klasse passiert über den

Abbildung 4.17: Klassendiagramm der LBS Komponente



Wert des ServiceEnums. Beispielsweise gibt der Aufruf von ServiceEnum.GOOGLE_PLACES die Klasse der GooglePlaces Implementation des I_Service Interfaces zurück, welche dann in der getService Methode Instanziiert wird. Jeder Service erhält bei der Instanziierung den

Applikationskontext, da sie die konkrete Kommunikation zu den externen LocationBasedServices erledigen und für Internetaufrufe oder Änliches eben diesen Kontext benötigen könnten. Eine Suchanfrage ist gekapselt in einem ServiceQuery Objekt und gibt über den Aufruf der search Methode des konkreten LocationBasedServices eine Liste von ServiceResults zurück. Jedes ServiceResult enthält eine für den LocationBasedService eindeutige ID, die man benutzen kann, um den Service über Detailinformationen zu dieser ID zu befragen.

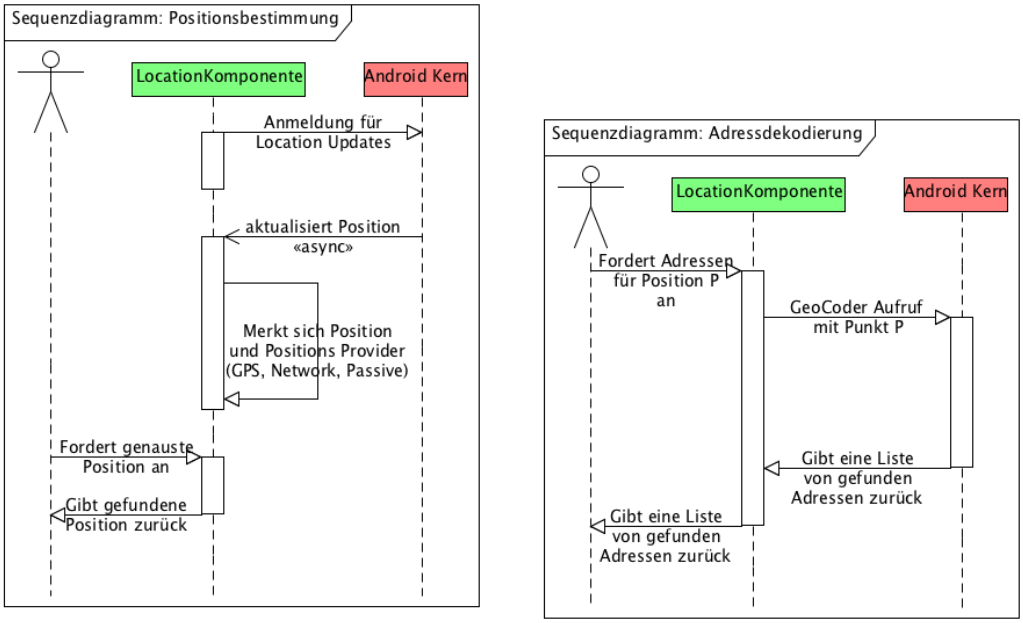
4.6 Sequenzdiagramme

Um einen Einblick in den Workflow der Applikation zu erlangen werden im Folgenden die wichtigsten Abläufe als Sequenzdiagram dargestellt. Zusätzlich werden diese Diagramme mit Screenshots der Prototyp Applikation angereichert, um das Erscheinungsbild dieser Applikation einzufangen.

4.6.1 Positionsbestimmung und Adressdekodierung

Die folgenden beiden Sequenzdiagramme sollen verdeutlichen, in wie weit ein Aufruf bei der Location Komponente eine Zusammenarbeit mit dem Android Kern auslöst. Bei der

Abbildung 4.18: Sequenzdiagramme: Positionsbestimmung und Adressdekodierung

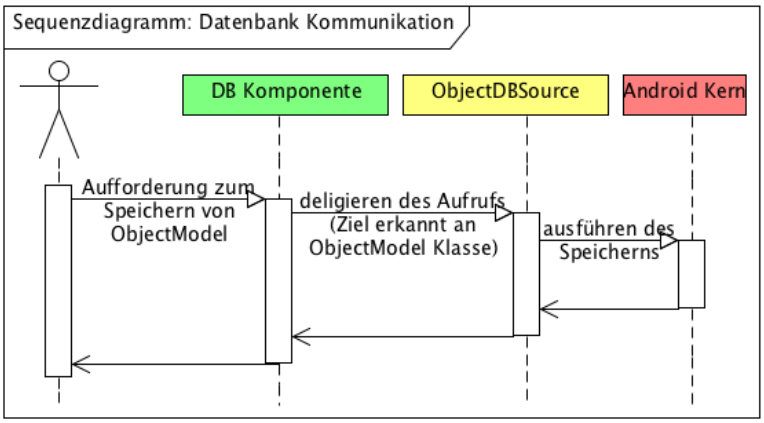


Positionsbestimmung sieht man, dass die Zusammenarbeit, wie im Kapitel 4.4.2 beschrieben, schon vor dem eigentlichen Aufruf passiert. Hier erhält die Komponente Informationen über Positionsänderungen. Die Adressdekodierung hingegen ruft den GeoCoder, welcher Teil des Android Kerns ist, synchron auf. Die weiteren Diagramme werden diese Einzelheiten nicht beinhalten, um die Komplexität zu reduzieren.

4.6.2 Kommunikation mit der Datenbank

Im folgenden Sequenzdiagram soll der Ablauf des Speicherns eines Objekts verdeutlicht werden. Dieser Ablauf ist representativ für alle Datenbank Kommunikationen über die Datenbank Komponente, da sich ausschließlich die Parametrisierung der Aufrufe und der Rückggabe Wert ändern. Da die Klasse der Zielentität des Aufrufs jedoch immer in den Methodensignaturen der Datenbank Komponente vorhanden ist, kann dieses Diagramm exemplartisch für alle Entitäten und Aufrufe gelten. (Hierzu gehören das Speichern, das Ändern oder das Löschen von Daten)

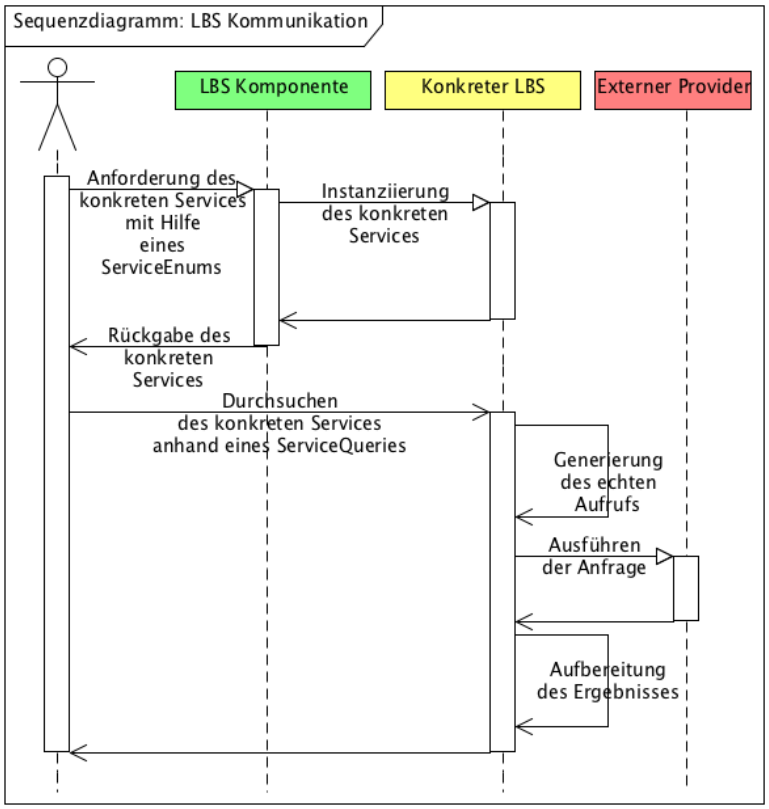
Abbildung 4.19: Sequenzdiagramme: Kommunikation mit der Datenbank



4.6.3 Kommunikation mit Locationbased Services

Das folgende Sequenzdiagramm zeigt den Ablauf des Durchsuchens eines Locationbased Services. Dieser Ablauf ist repräsentativ für das Durchsuchen und für Detailanfragen an den Service, da sich einzig der Query unterscheidet.

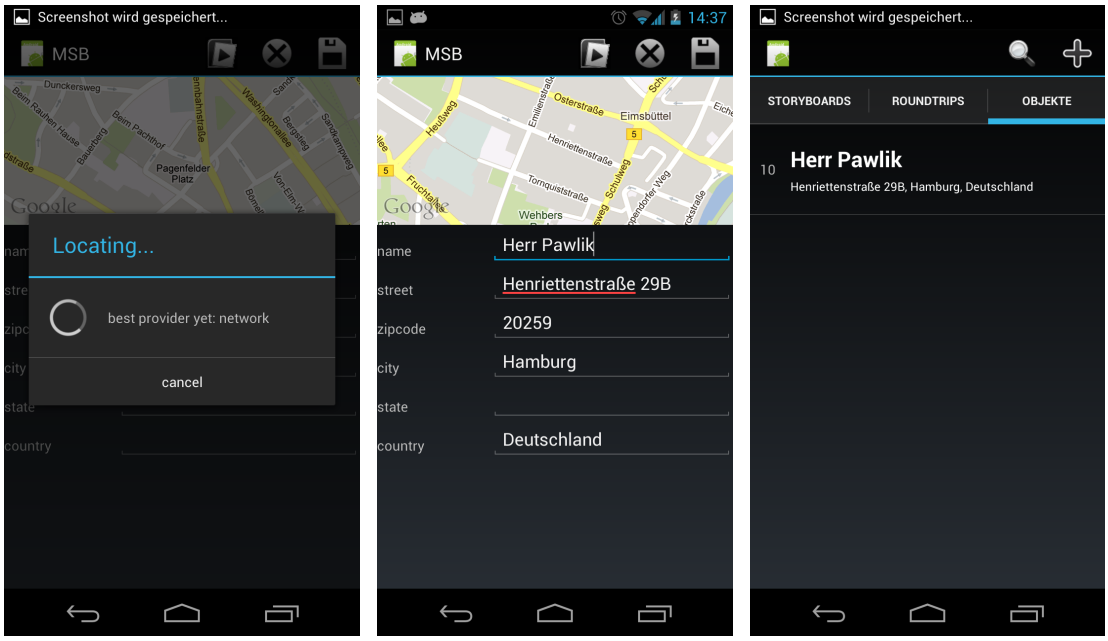
Abbildung 4.20: Sequenzdiagramme: Kommunikation mit LBS



4.6.4 Objekt erstellen

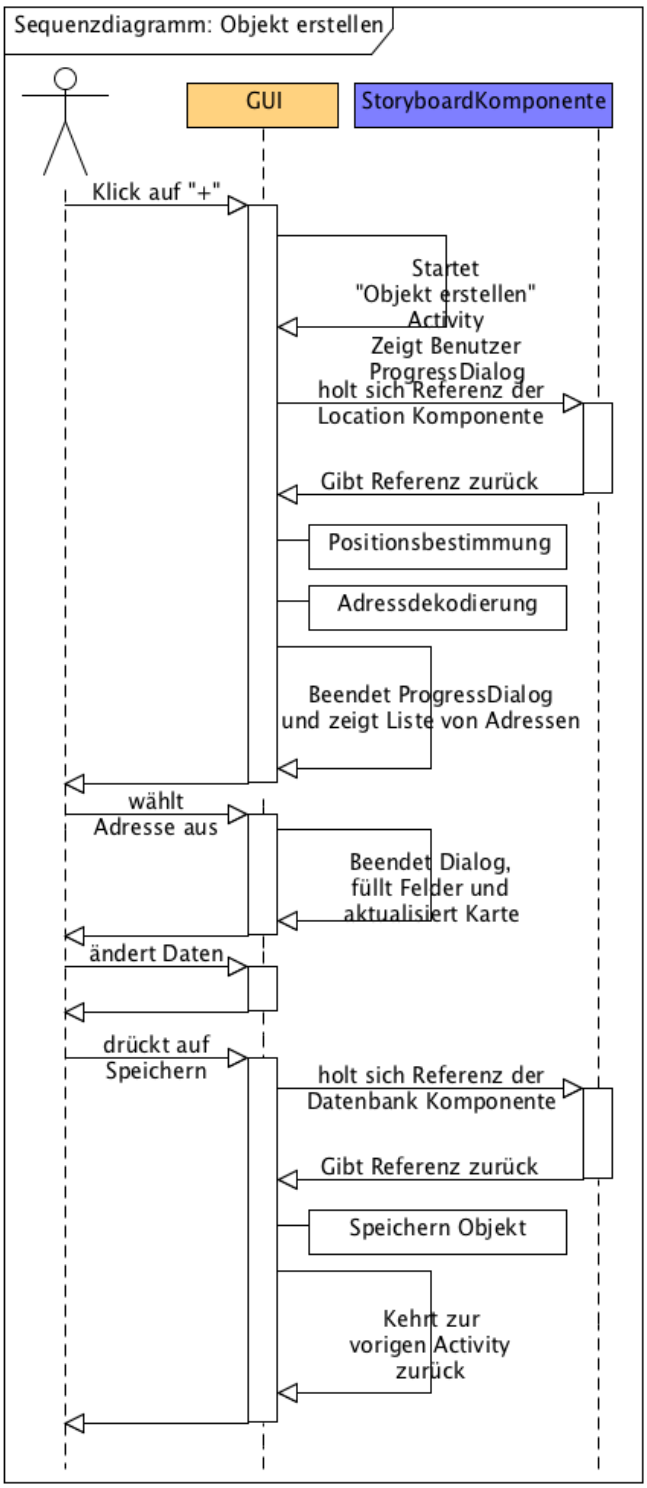
Der Ablauf des Objekt Erstellens wird im Sequenzdiagramm 4.6.4 dargestellt. Hierbei wurden die Teile, die in den vorherigen Sequenzdiagrammen dargestellt wurden, vereinfacht um die Komplexität des Diagramms zu reduzieren. Die beigefügten Screenshots sollen den Ablauf in der GUI analog zum Sequenzdiagramm darstellen.

Abbildung 4.21: Screenshots: Objekt erstellen



Startet der Benutzer das Erstellen eines neuen Objektes, öffnet sich die 'Objekt erstellen' Activity, welche nach dem Öffnen nach der aktuellen Position des Benutzers sucht. Nach Erhalt der aktuellen Position (1) werden alle Felder vorgefüllt und können vom Benutzer editiert werden (2). Durch Tippen auf das Speichern-Symbol, gelangt der Benutzer in das Hauptmenü zurück, indem das neue Objekt nun mit aufgelistet wird (3).

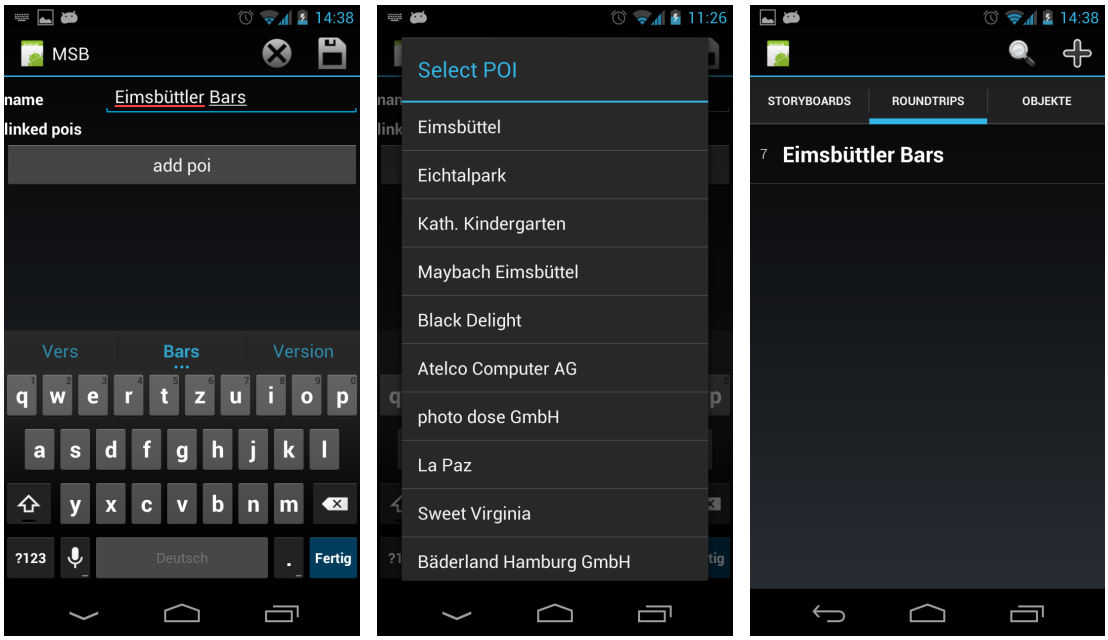
Abbildung 4.22: Sequenzdiagramm: Objekt erstellen



4.6.5 Roundtrip erstellen

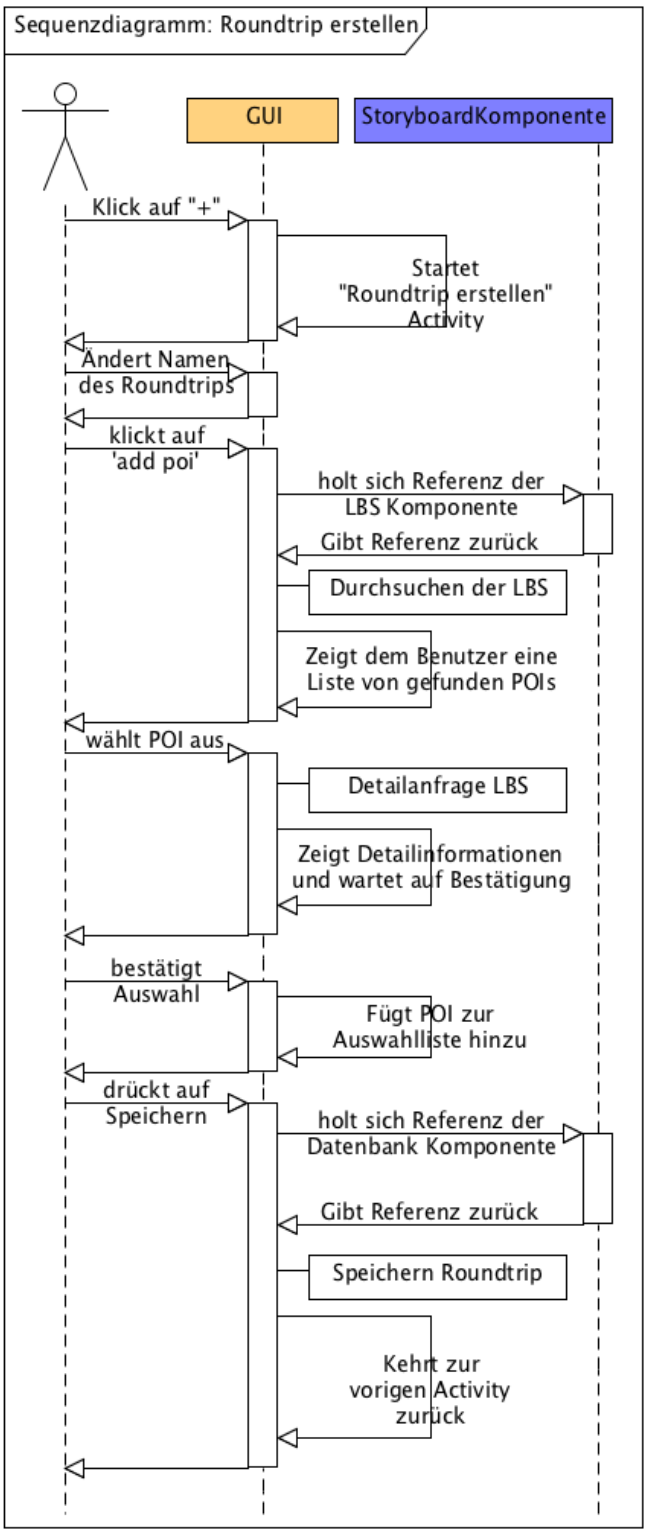
Das Sequenzdiagramm 4.6.5 zeigt den Ablauf des Roundtrip Erstellens. Auch in diesem Diagramm wurden Teile vereinfacht um die Komplexität zu veringern. Auf einzelne Screenshots musste an dieser Stelle verzichtet werden, da sich dieser Teil noch in der Entwicklung befindet.

Abbildung 4.23: Screenshots: Roundtrip erstellen



Nach dem Öffnen der 'Neuen Roundtrip erstellen' Activity, gibt der Benutzer einen Namen für den neuen Roundtrip ein (1). Der Klick auf den 'add poi' Button, führt eine Suche in den Locationbased Services aus und gibt eine Liste der gefunden POIs zurück (2). Der Benutzer wählt den gewünschten POI aus und fügt diesen, dem Roundtrip hinzu. Nach Druck auf das Speichern-Symbol, gelangt der Benutzer zurück in das Hauptmenü und findet seinen neu erstellten Roundtrip in der Liste vor (3).

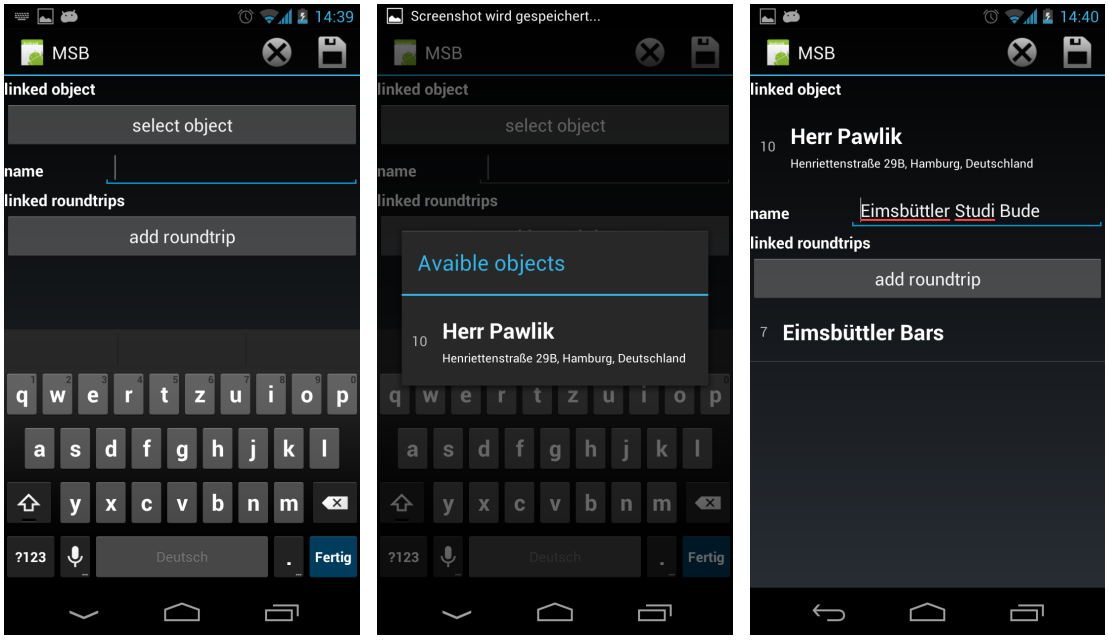
Abbildung 4.24: Sequenzdiagramm: Roundtrip erstellen



4.6.6 Storyboard erstellen

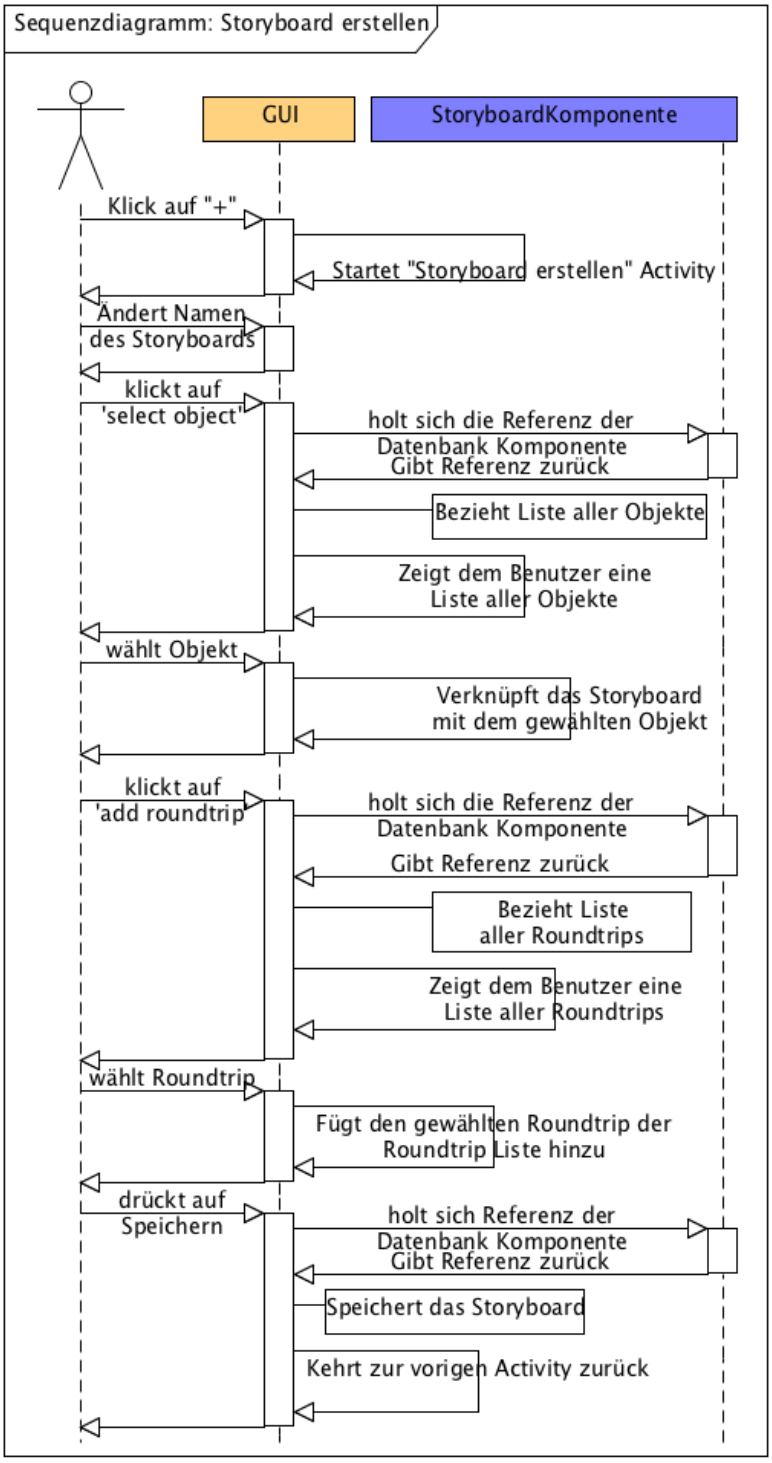
Im Diagramm 4.6.6 kann man den Ablauf des Storyboard Erstellens nachvollziehen. Das Verknüpfen mit einem Objekt sowie einem Roundtrip wird in den Screenshots gezeigt.

Abbildung 4.25: Screenshots: Storyboard erstellen



Fängt der Benutzer mit dem Erstellen eines neuen Storyboards an, so befindet er sich in der 'Storyboard erstellen' Activity (1). Hier verknüpft er das Storyboard mit einem Objekt über den 'select object' Button. Hierdurch wird ihm eine Liste der verfügbaren Objekte angezeigt (2). Durch das Klicken auf den 'add roundtrip' Button, wird ihm eine Liste der verfügbaren Roundtrips gezeigt, welche er durch Selektion mit dem Storyboard verknüpfen kann (3).

Abbildung 4.26: Sequenzdiagramm: Storyboard erstellen



4.7 Zusammenfassung

In diesem Kapitel wurde das Android Programmiermodell mit seinen Komponenten vorgestellt und der Bezug der Komponenten zur Arbeit hergestellt. Um zu einer wartbaren und erweiterbaren Software zu gelangen, stellt dieses Kapitel in 4.3 die in dieser Arbeit benutzten Entwurfsmuster vor. Der Vorstellung mit Hilfe der annotierten Komponentendiagramme folgt eine detailiertere Beschreibung der Klassen und Interfaces in 4.5. Der Nachweis der Erweiterbarkeit der Applikation wurde über die Realisierung der LBS Provider mit Hilfe des Factory Patterns bewerkstelligt. Durch Nutzung der Prinzipien der Android Komponenten Kommunikation (Intent / Intent-Filter) steht das hier entwickelte MobileStoryboard bestehenden Apps der Android Plattform (z.B.: Camera oder Gallery) unmittelbar zur Verfügung. Die Export Funktion sowie die Anbindung weiterer LBS-Provider wie YellowPages oder Qype sind bei der Prototypisierung ausgelassen worden, da diese den Rahmen dieser Arbeit gesprengt hätten.

5 Zusammenfassung und Ausblick

In dieser Arbeit wurde der Entwurf einer mobilen Applikation zur Unterstützung der Arbeit von Immobilienberatern erstellt. Mit Hilfe sogenannter Locationbased Services sollten Daten von Immobilien in Storyboards mit Umgebungsinformationen angereichert werden, um eine Immobilie besser als in einem Exposé präsentieren zu können. Im Kapitel 2 wurden hier zunächst existierende Applikationen, die ebenfalls Locationbased Services nutzen, untersucht. Wie in dem entsprechenden Kapitel beschrieben, haben die untersuchten Applikationen eher einen Suchcharakter, als dass sie dem Zweck einer Storyboard Applikation entsprechen. Im darauf folgenden Kapitel 3 wurde eine Analyse erstellt und Anwendungfälle sowie die Entitäten und Akteure einer MobileStoryboard Applikation definiert. Außerdem wurde der Begriff Locationbased Service erläutert, um im weiteren Verlauf der Arbeit ein Grundverständnis zu schaffen. Das Kapitel 4 befasste sich mit der Konzeption der MobileStoryboard Applikation. Im ersten Teil wurden dort die Anforderungen an die Soft- und Hardware definiert. Durch die weite Verbreitung, dem vertretenen OpenSource Gedanken und dem offenen Programmiermodell, ist die Entscheidung in diesem Kapitel auf das Android Betriebssystem als Grundlage dieser Arbeit gefallen. Die Konzepte und Komponenten von Android wurden mit Bezug auf diese Arbeit beschrieben. Weiter wurden die benutzten Entwurfsmuster im Kapitel 4.3 beschrieben, die der Lösung für das Mobile Storyboard zugrunde liegen. Im weiteren Verlauf wurden der Komponentenschnitt sowie der Softwareentwurf anhand von Klassen- und Sequenzdiagrammen erläutert.
In einem Prototypen des Entwurfs wurden die Erstellungsabläufe für Objekte, Roundtrips sowie Storyboards umgesetzt. Das Einbinden von Locationbased Services inklusive des Durchsuchens und der Detailabfragen sind ebenfalls Bestandteil des Prototyps. Storyboards können mit Objekten und Roundtrips verknüpft werden sowie Roundtrips mit POIs befüllt werden können. Für alle Entitäten der Applikation wurde mit Hilfe der Datenbank Komponente, unter Verwendung der SQLite Datenbank aus Android, die Persistenz realisiert. Die lose Kopplung an bestehende Kamera Software wurde ebnfalls implementiert. Der Prototyp wurde auf einem Galaxy Nexus der Marke Samsung mit installierter Android Version 4.0 (Icecream Sandwich) und 4.1 (Jelly Bean) getestet und für funktionstüchtig befunden.

Die Möglichkeit des Importierens und Exportierens von Storyboards ist nicht Bestandteil des Prototypen. Der Nachweis der Anbindung an externe Locationbased Service Provider wurde exemplarisch durch die Implementierung der Schnittstellen für Google Places erbracht. Die Anbindung weiterer Provider wie zum Beispiel YellowPages ist analog zu der im Prototyp gezeigten Lösung.

5.1 Ausblick

Als Erweiterung dieser Arbeit bietet sich die Import- und Exportfunktion an, durch die man zusammengestellte Storyboards zum Beispiel per E-Mail versenden oder über das FTP Protokoll auf entfernte Server hochladen könnte. Durch die im Kapitel 4.2 beschriebe Kommunikation zwischen den Applikation könnte man einen Import über Intent-Filter entwerfen, die auf entsprechende Aktionen, wie das Öffnen eines E-Mail Anhangs, reagieren. Wichtig bei dem Import und Export von Storyboards ist eine durchdachte Serialisierung und Deserialisierung von Storyboards, sowie der Umgang mit Dopplungen in der Datenbank und von Medien auf dem Dateisystem. Als zusätzliche Erweiterung bieten sich weitere Locationbased Service Provider an, die über die im Kapitel 4.4.3 und 4.5.4 beschriebene LBS Komponente leicht in die bestehende Applikation eingebettet werden können. Weiter sei hier die Möglichkeit der Erweiterung durch einen eigenen Locationbased Service genannt, der Roundtrips über Region of Intrest (ROI) georeferenzierbar macht. Ein solcher Service würde es Immobilienberatern ermöglichen, ihre Arbeit schneller und effizienter zu erledigen, indem sie, sofern andere Benutzer eines solchen LBS bereits Roundtrips erstellt haben, diese weiterverwenden können.

Literaturverzeichnis

[Deutschland 2012] Deutschland, Financial T.: Stückzahlen und Marktanteile nach Betriebssystemen. (2012), 07. – URL http://www.ftd.de/politik/konjunktur/:smartphones-stueckzahlen-und-marktanteile-nach-betriebssystemen/60130150.html

[Gamma;Helm;Johnson;Vlissides 2004] Gamma;Helm;Johnson;Vlissides: *Entwurfsmuster. Elemente wiederverwendbarer objektorientierter Software.* Addison-Wesley, München, 2004

[Google 2012a] Google: Activities. (2012), 07. – URL http://developer.android.com/guide/components/activities.html

[Google 2012b] Google: The Google Places API. (2012), 07. – URL https://developers.google.com/places/documentation/

[Google 2012c] Google: Intent and Intent-Filter. (2012), 07. – URL http://developer.android.com/guide/components/intents-filters.html

[Google 2012d] Google: Permissions. (2012), 07. – URL http://developer.android.com/guide/topics/security/permissions.html

[Google 2012e] Google: Services. (2012), 07. – URL http://developer.android.com/guide/components/services.html

[Stefan Steiniger 2006] Stefan Steiniger, Alistair E.: *Foundations of Location Based Services.* 2006

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 26. Juli 2012 Michael Pawlik