HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Studienarbeit, Abteilung Informatik

Android Applikation RadioTour

Hochschule für Technik Rapperswil

Frühjahrssemester 2012
01. Juni 2012

Autoren:	Florian Bentele & Daniel Stucki
Betreuer:	Prof. Dr. Peter Heinzmann
Projektpartner:	Lukas Frey, cnlab AG, Rapperswil-Jona
Arbeitsperiode:	21.02.2012 - 01.06.2012
Arbeitsumfang:	240 Arbeitsstunden bzw. 8 ECTS pro Student

Abstract

Bisher mussten bei Radrennen Motorradfahrer die Nummern der Fahrer, welche aus dem Feld ausgerissen sind, dem *RadioTour Speaker* melden. Dieser meldet dann die Ausreisser jeweils per Funk an die Mannschaftsleiter weiter. Bei den meisten Radrennen notieren die *RadioTour Speaker* die erhaltenen Informationen in ein Notizbuch, bevor sie die Fahrernummern weitergeben. An der Tour de Suisse wir seit einigen Jahren mit einer Tablet-PC Web Applikation zur elektronischen Erfassung der Renninformationen gearbeitet.
Die im Rahmen dieser Studienarbeit realisierte *RadioTour* Android Tablet Applikation ersetzt und erweitert die „in die Jahre gekommene“ webbasierte Tablet-PC Anwendung. Die neue native *RadioTour* Android Anwendung bietet eine einfachere Touchscreen Bedienung, verbesserte Importfunktionen sowie neue Features wie beispielsweise Live-Marschtabellen und Streckenkilometerbestimmung aus den lokalen GPS-Daten. Verbessert wurde auch die Kommunikation mit der TourLive Webseite via JavaScript Object Notation (JSON) zur unmittelbaren Veröffentlichung der aktuellen Rennsituationen. Die Anwendung ist bewusst für Android Honeycomb & Ice Cream Sandwich und für die spezifische Hardware-Plattform eines Samsung Galaxy Tab 10.1 entwickelt worden, da das System zusammen mit der Hardware-Plattform zur Verfügung gestellt wird.
Für die Entwicklung der Applikation kommt die Java Integrated Development Environment (IDE) Eclipse (Version Indigo) zur Anwendung. Die Datenpersistierung auf dem Tablet wird mithilfe der ORMLite Library (Version 4.39) in einer SQLite Datenbank umgesetzt.
Der Release 1 wurde an der Berner Rundfahrt 2012 getestet. Die dort gewonnenen Erkenntnisse wurden im Release 2 berücksichtigt, so dass nun ein System vorliegt, welches die Grundanforderungen erfüllt. Letzte, kleinere Anpassungen wird der Industriepartner im Rahmen der Systemübernahme vornehmen. Die neue *RadioTour* Anwendung wird voraussichtlich bei der Tour de Suisse 2012 zum Einsatz kommen.

Aufgabenstellung

Studiengang:	Informatik (I)
Semester:	FS 2012 (21.02.2012-01.06.2012)
Institut:	ITA: Internet-Technologien und Anwendungen
Gruppe:	Florian Bentele, Daniel Stucki
Verantwortlicher:	Dr. Prof. Peter Heinzmann, pheinzma@hsr.ch
Industriepartner:	cnlab AG, Lukas Frey, lukas.frey@cnlab.ch

Ausgangslage

Bei Radrennen erfasst der so genannte *RadioTour Speaker* Informationen zur Rennsituation, welche ihm von Motorradfahrern per Funk geliefert werden. Gegenwärtig legt der *RadioTour Speaker* mit Hilfe einer TabletPC Web-Anwendung per Click auf die erhaltenen Fahrernummern die Zusammensetzung der Gruppen fest. Er tippt auch die Zeitabstände zwischen den Gruppen ein. Die *RadioTour* Anwendung visualisiert die Fahrergruppen und liefert Detailinformationen zu den Fahrern (z.B. Namen, Team, virtueller Rang). Veränderungen in den Fahrergruppen können per Drag-and-Drop nachgeführt werden. Die mit der *RadioTour* Anwendung erfassten Rennsituationen werden per Mobilfunknetz zu einem Webserver gesendet, wo sie weiteren Anwendungen, z.B. Live Webinformationen zur Verfügung stehen.

Ziel

Die aus dem Jahre 2006 stammende Notebook Web Applikation soll nun dahingehend überarbeitet und erweitert werden, dass man sie auf Android-Tablets oder iPads betreiben kann.
Das Produkt wird spezifisch auf ein Gerät und nicht plattformübergreifend entwickelt. Die Verbindung zum Server wird in der Arbeit definiert, jedoch werden keine serverseitigen Entwicklungen erarbeitet.
Die Mehrsprachigkeit wird nach Android Standards implementiert [1]. Eine Übersetzung ist jedoch nicht Teil der Arbeit.

1. Android Internationalisierung, `http://developer.android.com/guide/topics/resources/localization.html`

Teilaufgaben

- Analyse der existierenden *RadioTour* und TourLive-Anwendungen
  - Tour de Suisse Dokumente
  - Alte *RadioTour* Anwendung
    `http://gps.cnlab.ch/tablet/`
    user: ba_tourlive
    password: access4tl
  - TourLive-System (GPS Positions- und Bilderfassungssysteme, Webanwendung)
  - Kommunikation *RadioTour* – TourLive-Webanwendung
  - Vergleich mit Systemen anderer Radrennen
- Festlegung der Funktionalität der neuen Tablet-Anwendung (Requirements Engineering)
  - Studium der Geschäftsprozesse (Renninformationen, Rennverlauf)
  - Austesten von Teilfunktionen (Android-Tablet Programmierung, GPS-Positionserfassung, Usability Experimente)
  - Erweiterte Funktionen (z.B. Erfassung der Streckenkilometer, Integration von Marschtabellen)
  - Auswahl der Hardware-Plattform
  - Spezifikation
- Design
  - Benutzerschnittstelle
  - Kommunikation mit TourLive Aufnahmesystemen (Weitergabe der Daten an Datenserver)
  - *RadioTour* Anwendung
- Realisierung
  - Prototypen
  - Anpassungen
  - Friendly User Test Version (Beta-Release)
  - Feldtest an einem Radrennen
  - Übergabe an cnlab
- Dokumentation
  - Gemäss Anforderungen Industriepartner (das System soll vom Industriepartner betrieben und erweitert werden können)
  - Bericht gemäss HSR / Heinzmann Richtlinien

Diese Aufgabenstellung wird genehmigt vom Betreuer

Ort / Datum P. Heinzmann

Erklärung zur Urheberschaft[2]

Die vorliegende Arbeit basiert auf Ideen, Arbeitsleistungen, Hilfestellungen und Beiträgen gemäss folgender Aufstellung:

Gegenstand, Leistung	Person	Funktion
Kapitel 3, 4, 5 und Anhänge	Daniel Stucki	Autor der Arbeit
Kapitel 1, 2, 5, 6 und Anhänge	Florian Bentele	Autor der Arbeit
Korrektur	Heinrich Stucki	Lektorat
Korrektur	Doris Bentele	Lektorat
Korrektur	Ursina Bentele	Lektorat
Idee, Aufgabenstellung, allgemeines Pflichtenheft, Betreuung während der Arbeit	Prof. Dr. P. Heinzmann	Verantwortlicher Professor
Industriepartner und Ansprechperson	Lukas Frey	cnlab AG

Ich erkläre hiermit,

- dass ich die vorliegende Arbeit gemäss obiger Zusammenstellung selber und ohne weitere fremde Hilfe durchgeführt habe,
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe.

Rapperswil, 29. Mai 2012

Florian Bentele Daniel Stucki

2. Diese Erklärung basiert auf der Muster-Erklärung in den Richtlinien der HSR zur Durchführung von Projekt-, Studien-, Diplom- oder Bachelorarbeiten vom 16. Februar 2009.

Vereinbarung zur Verwendung und Weiterentwicklung der Arbeit[3]

1. Gegenstand der Vereinbarung

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Studienarbeit „Android Applikation RadioTour“ von Florian Bentele und Daniel Stucki unter der Betreuung von Prof. Dr. Peter Heinzmann (für die Arbeit verantwortlicher Professor) geregelt.

2. Urheberrecht

Die Urheberrechte stehen der Studentin / dem Student zu.

3. Verwendung

Die Ergebnisse der Arbeit dürfen sowohl von allen an der Arbeit beteiligten Parteien, d.h. von den Studenten, welche die Arbeit verfasst haben, vom verantwortlichen Professor sowie vom Industriepartner verwendet und weiter entwickelt werden. Die Namensnennung der beteiligten Parteien ist bei der Weiterverwendung erwünscht, aber nicht Pflicht.

Rapperswil, den

Florian Bentele

Rapperswil, den

Daniel Stucki

Rapperswil, den

Prof. Dr. Peter Heinzmann

Rapperswil, den

Industriepartner, Lukas Frey, cnlab AG

3. Diese Vereinbarung basiert auf den Muster-Vereinbarungen in den Richtlinien der HSR zur Durchführung von Projekt-, Studien-, Diplom- oder Bachelorarbeiten vom 16. Februar 2009.

Management Summary

Ausgangslage

An der Tour de Suisse fahren ca. 200 Radrennfahrer in Tagesetappen durch die ganze Schweiz. Dabei werden Sie von diversen Motorfahrzeugen begleitet. Im Radfahrer-Feld fährt ebenfalls der *RadioTour Speaker* mit. Seine Funktion besteht darin, Live Informationen des Rennens zu erfassen und an den Server der cnlab AG weiterzuleiten. Die Übertragung der Daten vom Gerät zum Server geschieht über das Mobilfunknetz 3G.

Live Informationen

Während dem Rennen werden aus verschiedenen Quellen Informationen gesammelt. Zum einen sind dies Veränderungen im Rennfeld, zum anderen Wertungen, welche die Fahrer erreichen können, so z.B. einen Bergsprint. Diese Daten werden vom *RadioTour Speaker* manuell erfasst.
Wenn sich ein Rennfahrer vom Feld ablöst und einen Vorsprung erarbeitet, wird dieser von einem Motorradfahrer verfolgt. Diese Änderung wird dann sofort per Funk dem *RadioTour Speaker* übermittelt.

Äussere Bedingungen

Bei Live Sport Events wie der Tour de Suisse ist die Erfassung von Echtzeitdaten, aus technischer Sicht, eine Herausforderung. Sowohl das alpine Gebirge, wo die Mobilfunkverbindungen und GPS Informationen nicht immer gewährleistet sind, als auch die ständige Vibration der Fahrzeuge stellen erschwerende Rahmenbedingungen dar. Die Unterbrüche der Verbindung werden überbrückt, indem die Änderungen gesammelt und periodisch an den Server gesendet werde.

Vorgehensweise

In der vorliegenden Studienarbeit kommt das Vorgehensmodell zur Softwareentwicklung von Rational Unified Prozess (RUP) zur Anwendung. Das Projekt wird in die folgenden vier Phasen aufgeteilt:

1. Inception
2. Elaboration
3. Construction
4. Transition

In jeder dieser Phasen werden die Arbeitsschritte nach RUP durchgeführt, je nach dem in mehreren Iterationen, wie es bei dieser Arbeit in der Phase *Construction* vorkommt.[4]
Die Erfassung der Anforderungen, der Entscheid zur Entwicklung auf einem Android Gerät sowie die Evaluation eines geeigneten Tablets bilden zusammen die Startphase des Projekts. Die Anforderungskriterien an das Tablet wurden in einer Sitzung zusammen mit Herrn Dr. Prof. Peter Heinzmann, dem Betreuer der Arbeit, diskutiert und genehmigt.
Im weiteren Verlauf der Arbeit wurden die Anforderungen und die UseCases definiert. Daraus entstand dann die Domainlogik und parallel dazu ein erster Prototyp des UserInterface. Insbesondere die Benutzerschnittstelle entstand in mehreren Iterationen, da eine gute Bedienung für den Erfolg des Produktes entscheidend ist, dies jedoch erst bei der realen Anwendung geprüft werden kann.

Ergebnisse

Die *RadioTour* Android Applikation beinhaltet die festgelegten Anforderungen. Die Fahrerlisten und die offiziellen Zeitmessungen können via USB oder aus dem Internet importiert werden. Die Gruppen lassen sich dynamisch verändern und die Rennsituation wird an den Server übermittelt. Ein Testlauf mit dem ersten Prototypen hat klar aufgezeigt, dass diese Anwendung eine Verbesserung in der Bedienung bringt. Dabei entstehen keine Einbussen in der Funktionalität. Die Applikation ist damit für den Einsatz an der Tour de Suisse bereit.

4. Wikipedia, Rational Unified Process, `http://de.wikipedia.org/wiki/Rational_Unified_Proces`, aufgerufen am 20.05.2012.

Inhaltsverzeichnis

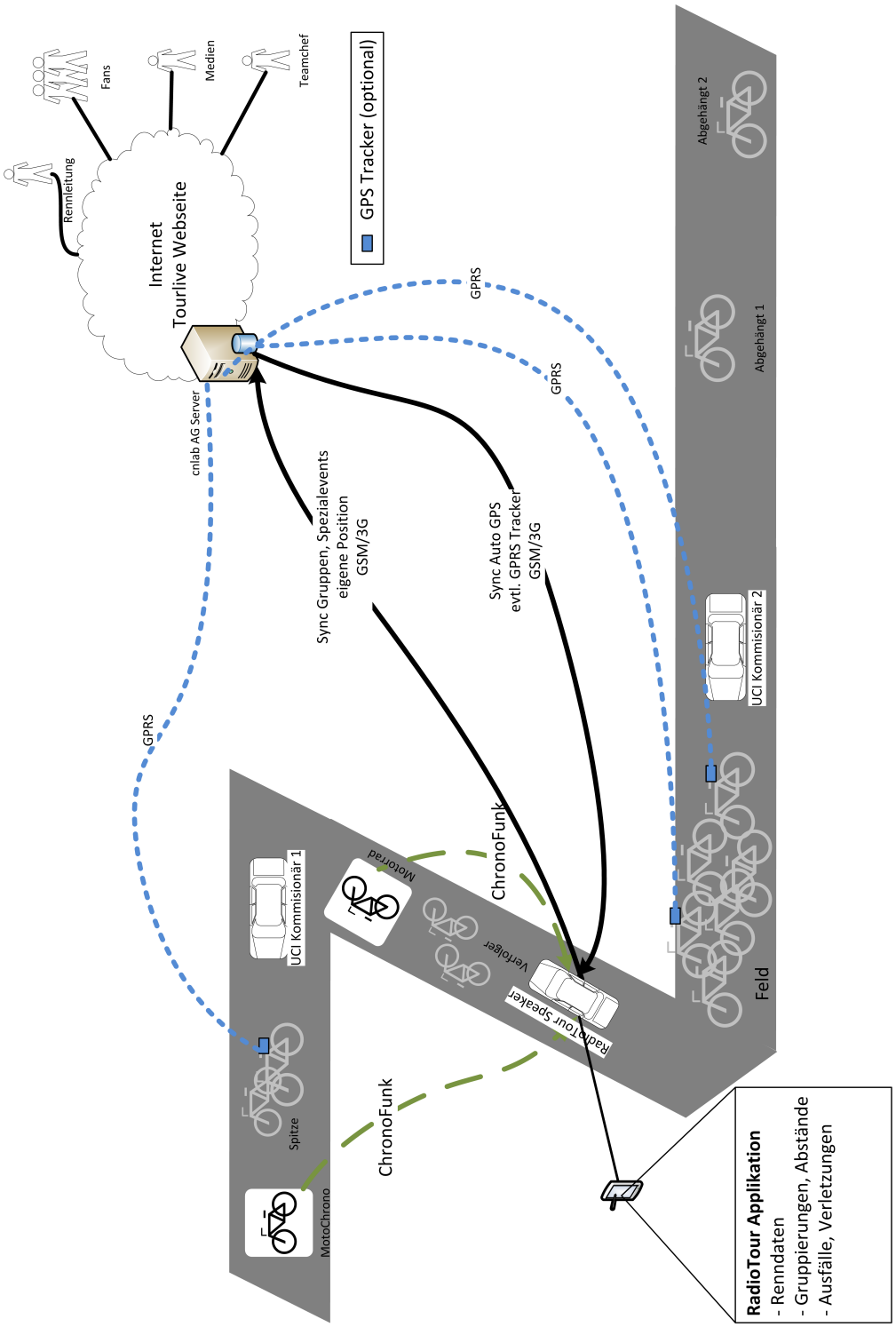
1 Einleitung

Im folgenden Abschnitt werden die aus technischer Sicht relevanten Aspekte genauer analysiert. Zu Beginn wird das Aufgabenumfeld in einem weiteren Sinne betrachtet, und die Kernelemente der Applikation aufgezeigt. Danach wird auf die Analyse und die Realisierung eingegangen.
Der Hauptteil richtet sich vor allem an Personen, die bereits Hintergrundwissen zu Android vorweisen, sowie für Entwickler, die an der Weiterentwicklung des Produktes interessiert sind.

1.1 BigPicture

Zur Übersicht wird das Umfeld der Applikation in einem BigPicture zusammengefasst. Es ermöglicht die Darstellung der äusseren Einflussfaktoren sowie die Abgrenzung des Systems zu definieren.
Die schematische Darstellung zeigt im wesentlichen die drei Hauptakteure auf. Zum einen sind dies die Motorradfahrer, welche die Radrennfahrer begleiten und Veränderungen in Echtzeit per Funk übermitteln. Diese Informationen kommen in kurzen Abständen und müssen sofort erfasst werden. Im UserInterface wird dafür eine Lösung verwendet, bei der mehrere Radrennfahrer gleichzeitig eingetragen werden können.
Eine weitere Rolle spielt der *RadioTour Speaker* mit dem Android Tablet. Er fasst die Informationen zusammen und wertet diese bereits bei der Eingabe auf dem Gerät aus. Im Tablet werden auch Daten wie z.B. die Durchschnittsgeschwindigkeit und die aktuelle Rennzeit angezeigt.
Der dritte Akteur bildet der Server der cnlab AG, welcher direkt mit der Applikation kommuniziert. Ausgetauscht werden die Veränderungen im Feld sowie Rückstände von der Spitze. Weiter können Ereignisse wie z.B. eine Verletzung oder ein defektes Fahrrad aufgezeichnet werden. Die Daten werden dann weiter auf der Webseite der *TourLive* aufbereitet und publiziert. Nicht nur für die Beteiligten im Team, sondern auch für Fans sind diese Angaben von grossem Interesse, da die Daten vor den offiziellen Zeitmessungen bereits einen Einblick in das Schlussklassement geben.

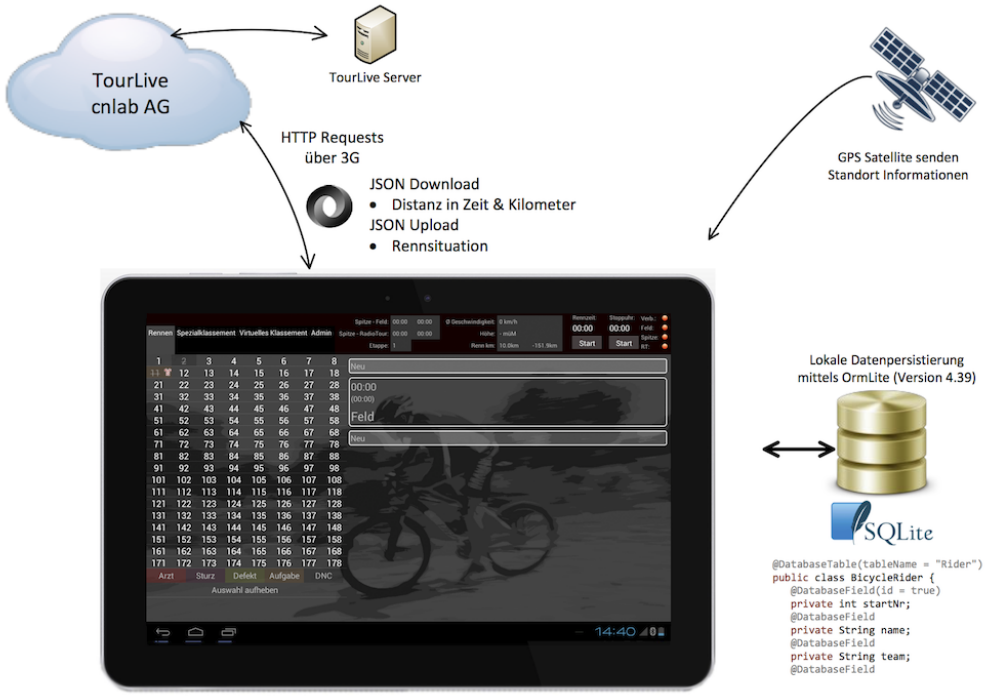
Abbildung 1.1: Das Aufgabenumfeld in einem BigPicture zusammengefasst



1.2 Kernelemente

Das Produkt dieser Arbeit ist es, die Tabletanwendung für den *RadioTour Speaker* zu entwickeln. Der Hauptfokus liegt dabei auf der Erfassung der Rennsituation und die sichere Übertragung an den Server. In der Abbildung 1.2 steht die Applikation im Mittelpunkt und zeigt die Hypertext Transfer Protocol (HTTP) Schnittstelle zum TourLive Server. Die Positionsdaten werden von den GPS Satelliten empfangen und die Datenpersistierung basiert auf einer SQLite Datenbank. In der Applikation ist die Hauptansicht für die Situation während einem Rennen zu sehen.

Abbildung 1.2: Die genaue Betrachtung der Aufgaben der Applikation



2 Analyse

Die Analyse untersucht die bestehende Web Applikation und fasst die Anforderungen zusammen. Danach wird die Plattform und die passende Hardware dazu evaluiert und in Form einer Kaufempfehlung an die cnlab AG weitergeleitet. Es werden die verwendeten Technologien sowie ein kurzer Exkurs zu anderen Lösungen angesprochen.

2.1 Requirements

Die Anforderungen an die *RadioTour* Applikation ergeben sich aus den Features der bisherigen Web Applikation und den Verbesserungsvorschlägen des *RadioTour Speakers*. Im Anhang 7.4 sind sämtliche UseCases der bisherigen Applikation aufgeführt. Im folgenden Abschnitt werden die wichtigsten Funktionen, welche in diesem Projekt implementiert sind, aufgeführt.

2.1.1 Funktionale Anforderungen

Die funktionalen Anforderungen sind in drei Prioritätsstufen eingeteilt:

zwingende Anforderungen (must)

- Fahrerliste, Etappen und Marschtabellen importieren
- Fahrer auflisten, sortieren und bearbeiten
- Gruppen bilden und Rückstand angeben
- Gruppen auflösen
- Events für Fahrer erfassen (Sturz, Arzt, Aufgabe)
- Rennsituation an den Server übermitteln
- Spezialklassemente und Wertungen erstellen und Fahrer zuweisen
- Maillots erfassen und bearbeiten
- Persistierung der Daten auf dem Gerät

optionale Anforderungen (can)

- Aktuelle Rennkilometer und Rennzeit anzeigen
- Stoppuhr

- Aktuelle Position durch GPS bestimmen
- Events für Fahrer an den Server übermitteln

wünschenswerte Anforderungen (nice to have)

- Aktuelle Position des *RadioTour Speakers* in der Marschtabelle anzeigen
- Splashscreen beim Start der Applikation

2.1.2 Nicht funktionale Anforderungen

Im Umfeld der *RadioTour* Applikation spielen neben den funktionalen Anforderungen an ein Software Produkt auch nicht funktionale Anforderungen eine wichtige Rolle. So muss z.B. das UserInterface Fehler bei der Eingabe tolerieren bzw. korrigierbar machen. Die Bedienung muss flüssig verlaufen und zu jedem Zeitpunkt muss der Status der Applikation sichtbar sein. Die für *RadioTour* relevanten nicht funktionalen Anforderungen sind im Folgenden aufgelistet.

- Mobilfunkverbindung ist nicht immer gewährleistet, keine Daten dürfen dadurch verloren gehen
- Informationen müssen auch bei direktem Sonnenlicht gut lesbar sein
- Angenehme, flüssige und selbsterklärende Bedienung der Applikation

Weitere nicht funktionale Anforderungen beziehen sich auf die Hardware an sich und sind im Kapitel 2.2 aufgeführt.

2.2 Evaluation und Kaufempfehlung

Die Evaluation der Zielplattform stellt einen wichtigen Faktor für die weitere Entwicklung der Arbeit dar. Zur Auswahl stehen die beiden marktführenden Betriebssysteme Android (Google) und iOS (Apple). Als Grundlage für die Evaluation dienen die folgenden Kriterien:

- Vorkenntnisse der Programmiersprachen Java bzw. Objective-C
- Möglichkeiten zum UserInterface Design
- Programmierumgebung, IDE
- Mögliche Vertriebskanäle der Applikation
- Nutzbarkeit von externen Geräten und Schnittstellen
- Vielfalt von Informationsquellen im Internet

Die Kriterien werden in einer Nutzwertanalyse gewichtet und bewertet. Insbesondere die Vorkenntnisse in Java sind ausschlaggebend für den Entscheid, die Applikation für die Androidplattform zu entwickeln. Dieser Entscheid ist in Absprache mit Herrn Heinzmann getroffen worden. Die gesamte Liste der Kriterien mit der jeweiligen Gewichtung sowie eine ausführliche Erläuterung befinden sich im Anhang 7.3.

Für die Auswahl eines geeigneten Tablets wird im nächsten Schritt ein Kriterienkatalog definiert mit zwingenden und optionalen Kriterien für das Gerät. Die zwingenden Kriterien beinhalten:

- Android Betriebssystem, gemäss Evaluation
- USB Anschluss für den Import der Fahrerliste am Renntag, optional auch mit Adapter möglich
- Mobilfunktnetz 3G für die Kommunikation mit dem Server
- GPS für die Lokalisierung
- Stromversorgung durch 12V (Auto) Adapter möglich

Zu den optionalen Kriterien gehören die Akkulaufzeit, falls die Stromversorgung unterbrochen wird, sowie ein grosszügiger Bildschirm für die Bedienung mit dem Finger oder mithilfe eines Stifts.

Als Sieger und somit auch als Kaufempfehlung an die *cnlab AG* geht das Lenovo ThinkPad Tablet. Dieses Gerät erfüllt alle Kriterien und überzeugt in der Vielfalt der Anschlüsse. Die Kaufempfehlung mit weiteren Erläuterungen ist im Anhang 7.3 zu finden.

Im Verlauf der Arbeit ist ein Defekt an der Micro USB Buchse entstanden. Dieser Anschluss wird für die Entwicklung auf dem Gerät dringend benötigt. Für die weitere Entwicklung ist ein Ersatzgerät angeschafft worden. Dabei handelt es sich um das Galaxy Nexus Tab 10.1[1]

2.3 Technologien

2.3.1 Android

Die native Programmiersprache für das Android Betriebssystem ist Java. Die Programmierung in Java bringt den Vorteil, dass auf die gesamte Application Programming Interface (API) von Android zugegriffen werden kann. Weiter sind die Geräte genau dafür ausgelegt und eine optimale Performance kann erreicht werden. Sämtliche Komponenten dieser Arbeit sind in Java geschrieben.

2.3.2 Externe Libraries

Android beinhaltet bereits ein umfangreiches Framework zur Entwicklung. Deshalb nutzt die *RadioTour* Applikation folgende zwei externe Libraries:

ORMLite[2] ist eine OpenSource Java Library, welche das Object-relational mapping (ORM) übernimmt. ORMLite bietet eine speziell auf Android angepasste Distribution. Um eine Klasse mit ihren Feldern zur Persistierung zu markieren, werden Java Annotationen verwendet. Aus diesen Annotationen erstellt ORMLite die Datenbank. Darüber hinaus werden mit ORMLite alle Zugriffe auf die SQLite Datenbank ausgeführt. In dieser Arbeit wird die ORMLite Version 4.39 verwendet.

Robotium[3] ist ein Test Framework, welches unter der Apache License 2.0 veröffentlicht und zur freien Nutzung angeboten wird. Robotium untersützt das Testen der

1. Galaxy Nexus Tab 10.1, `http://www.samsung.com/ch/consumer/mobile-phone/tablets/tablets/GT-P7500UWDITV`, Aufgerufen am 23.05.2012.
2. ORMLite, `http://ormlite.com/`, aufgerufen am 23.05.2012
3. Robotium, `http://code.google.com/p/robotium/`, aufgerufen am 30.05.2012.

Applikation mithilfe von UI Aktionen. In dieser Arbeit wird die Version 3.2.1 von Robotium verwendet.

2.3.3 Entwicklungsumgebung

Die von Android empfohlene Entwicklungsumgebung ist Eclipse[4] mit einem Plugin zur Entwicklung von Android Applikationen. Auf der Entwicklerseite von Android steht dazu folgendes:

> *Android Development Tools (ADT) is a plugin for the Eclipse IDE that is designed to give you a powerful, integrated environment in which to build Android applications.*[5]

Eclipse ist eine weit verbreitete IDE und wird aktiv weiter entwickelt. Mit dem Plugin zusammen bildet sie eine solide Grundlage für dieses Projekt.

Damit die Android Applikation direkt auf dem Computer getestet werden kann, stellt Google einen Emulator zur Verfügung. Der Emulator ist allerdings auch als solcher zu betrachten, da die Bedienung nicht vergleichbar mit einem richtigen Tablet ist.

2.3.4 Android Version

Eine Anwendung wird für eine spezifische Android Version entwickelt und getestet. Somit kann garantiert werden, dass das Verhalten der Anwendung immer gleich ist. In dieser Arbeit ist dies die Version 3.1 mit dem Versionsnamen *Honeycomb*.[6]. Weiter läuft die Applikation ebenfalls mit der Nachfolgeversion *IceCreamSandwick*.

Die Entwicklung auf einer Version schliesst jedoch nicht aus, dass die Anwendung in neueren Versionen nicht mehr lauffähig ist. Auch *RadioTour* kann für zukünftige Versionen weiterentwickelt und verwendet werden.

2.4 Mitbewerberanalyse

Die Art der Applikation ist sehr spezifisch und kann nicht direkt für andere Sportereignisse angewendet werden. Deshalb beinhaltet die Analyse von Mitbewerbern nur die grossen europäischen Radrennen. Wie bei der Tour de Suisse ist auch in Frankreich an der *Tour de France*[7] ein *RadioTour Speaker* mit dabei. Darüber, wie die Aufzeichnungen in Frankreich im genauen stattfinden, kann aber nur spekuliert werden, da die Informationen nicht öffentlich zugänglich sind.

In Italien findet zum Zeitpunkt dieser Arbeit der *Giro d'Italia*[8] statt. Bei diesem Radrennen ist es möglich, aus den Informationen, welche auf der Webseite verfügbar sind, zu schliessen, dass ein ähnliches System verwendet wird. Während dem Rennen ist es möglich, die aktuelle Rennsituation zu betrachten.

4. Eclipse, `http://eclipse.org/`, aufgerufen am 11.05.2012.
5. Android Plugin für Eclipse, `http://developer.android.com/sdk/eclipse-adt.html`, aufgerufen am 01.05.2012.
6. Android Honeycomb, `http://de.wikipedia.org/wiki/Android_(Betriebssystem)`, aufgerufen am 11.05.2012.
7. Tour de France, `http://www.letour.fr/`, aufgerufen am 01.05.2012.
8. Giro d'Italia, `http://www.gazzetta.it/Speciali/Giroditalia/2012/`, aufgerufen am 01.05.2012.

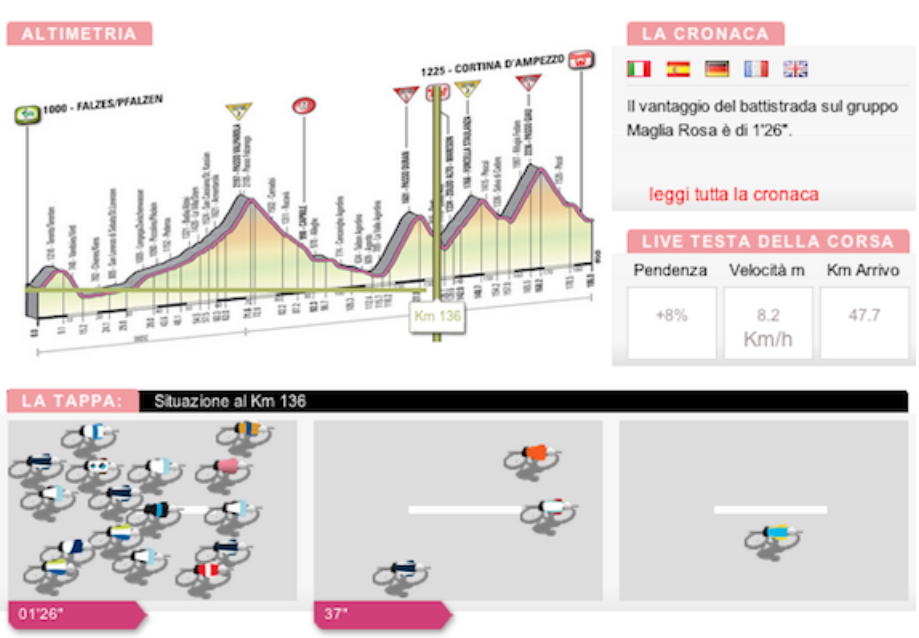


Abbildung 2.1: Rennsituation am Giro d'Italia

In der Abbildung 2.1 ist der Live Abschnitt der offiziellen Webseite zu sehen. Im oberen Teil wird der Standort in der aktuelle Etappe eingeblendet. Unten ist die Situation an der Spitze abgebildet. Die Fahrer sind nach Rückstand gruppiert.

Da jedoch nicht zu erkennen ist, wie die Informationen erfasst werden, muss die Mitbewerberanalyse an dieser Stelle abgeschlossen werden.

3 Architektur

Im folgenden Abschnitt wird die Architektur der Applikation diskutiert. Die Architektur ist so gewählt, dass die einzelnen funktionalen Komponenten zueinander eine tiefe Abhängigkeit aufweisen und dadurch eine weitere Entwicklung möglichst einfach ist.

3.1 Struktur der Applikation

Die Applikation hat im Grunde zwei Status, einerseits werden vor dem Rennen die Fahrerliste und die Marschtabelle importiert, andererseits wird die Rennsituation während dem Rennen erfasst und Änderungen festgehalten. Diese beiden Status können aber nicht absolut voneinander getrennt werden, da während dem Rennen Änderungen denkbar sind. Während dem Rennen müssen gewisse Daten immer angezeigt werden. Diese Live Informationen werden deshalb als eigene Ebene abgebildet. Aus den Anforderungen und den Kriterien entsteht folgende baumartige Struktur.

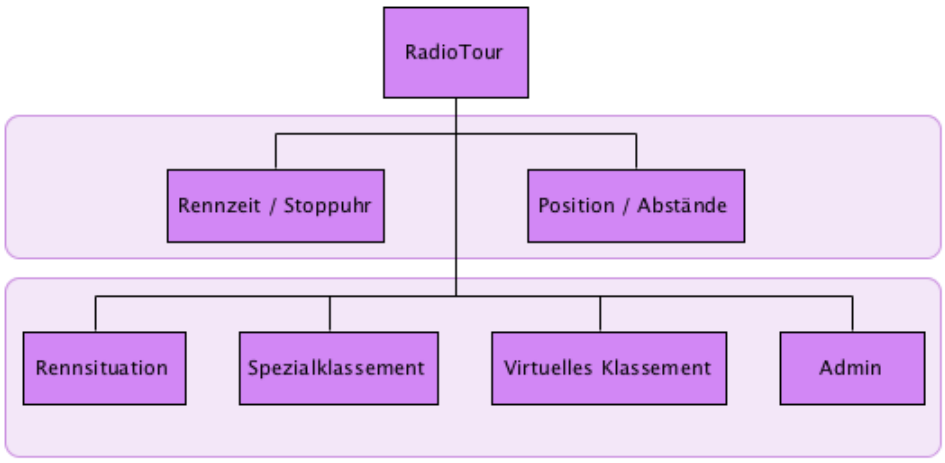


Abbildung 3.1: Struktur der Applikation als Organigramm

Der Stamm stellt die Applikation dar und die Äste zeigen die Aufteilung der Funktionen. Die Rennzeit sowie die aktuelle Rennposition sind in einem immer sichtbaren Bereich platziert.
Die untere Ebene beinhaltet die Kernelemente der Applikation. Diese werden in Views seitenweise dargestellt. Durch eine Navigation lässt sich zwischen den Views wechseln, ohne dass dabei die Live Informationen ausgeblendet werden.

3.2 Schichtenmodell und Paketdiagramm

Die Applikation lässt sich in vier verschiedene Schichten aufteilen. Diese Schichten sind in der Abbildung 3.2 illustriert. Die oberste Schicht stellt dabei die Schnittstelle zum Benutzer dar. Innerhalb der Schicht werden die dabei verwendeten Pakete angezeigt.

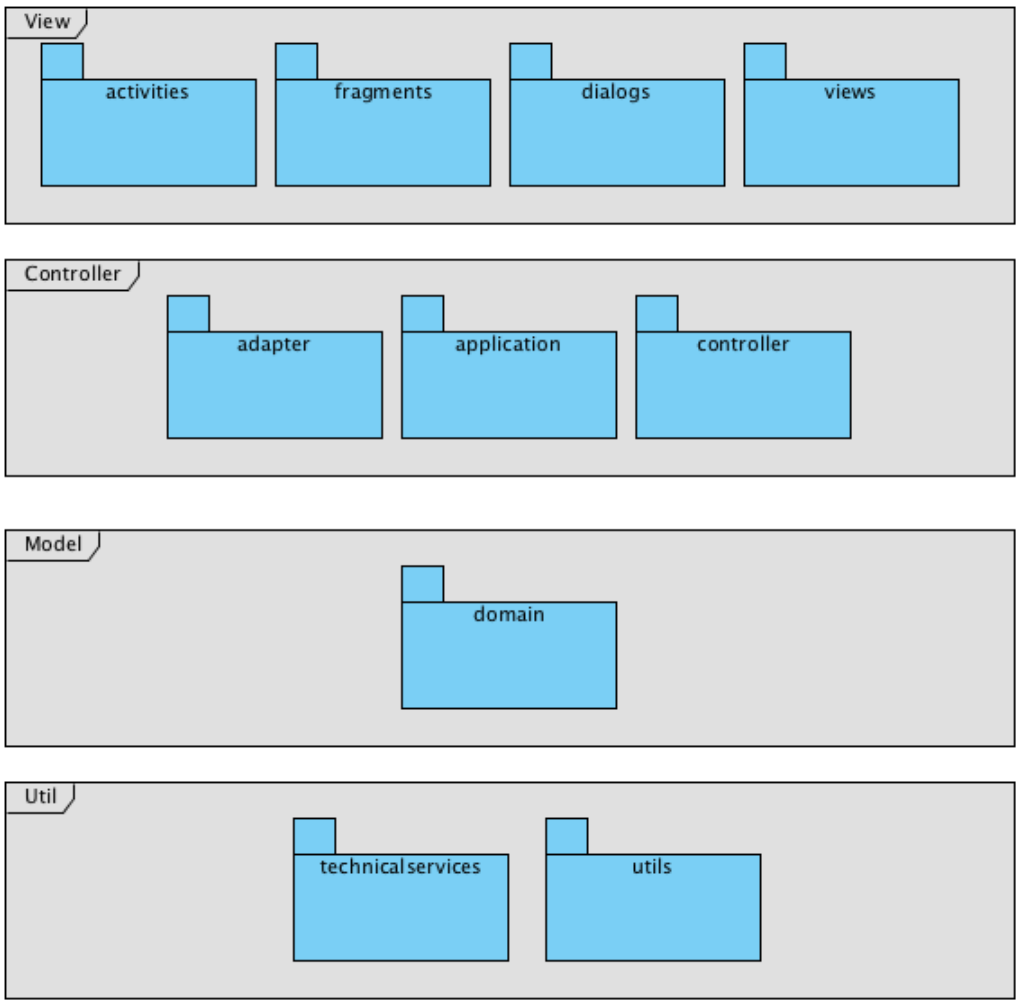


Abbildung 3.2: Die Schichten der Applikation inklusive der verwendeten Pakete

Die *View* Schicht in der Abbildung 3.2 enthält alle Pakete in welchen Elemente zur Benutzerinteraktion definiert sind. Um die einzelnen Pakete in dieser Schicht zu verstehen, wird hier ein kurzer Exkurs zu Android View Elementen eingeschoben. Für die Elemente *Activity* und *Fragment* wird hierfür ein Teil der Android API Beschreibung zitiert.

> *An Activity is an application component that provides a screen with which users can interact in order to do something, such as dial the phone, take a photo, send an email, or view a map. Each activity is given a window in which to draw its user interface. The window typically fills the screen, but may be smaller than the screen and float on top of other windows.*[1]

Die *RadioTour* Applikation besteht aus einer Activity. Dies aus der Entscheidung

1. Android Developers Reference, `http://developer.android.com/guide/topics/fundamentals/activities.html`, aufgerufen am 31.05.2012.

heraus, dass gewisse Inhalte, wie zum Beispiel die aktuelle Etappe oder die Stoppuhr jederzeit verfügbar sein müssen. Diese Activity befindet sich im Paket *activities*

> *A Fragment represents a behavior or a portion of user interface in an Activity. You can combine multiple fragments in a single activity to build a multi-pane UI and reuse a fragment in multiple activities. You can think of a fragment as a modular section of an activity, which has its own lifecycle, receives its own input events, and which you can add or remove while the activity is running (sort of like a „sub activity" that you can reuse in different activities).*[2]

Eine spezielle Form von Fragmenten stellen DialogFragmente dar. Sie werden benutzt um einen Dialog über der Activity einzublenden. In der Applikation werden diese Elemente verwendet um Daten zu erfassen und zu ändern. Die DialogFragmente der Applikation befinden sich im Paket *dialogs*.

Für eine genauere Analyse der in der *View* Schicht verwendeten Elemente und Klassen sei auf das Kapitel 4 verwiesen.
Die *Controller* Schicht in der Abbildung 3.2 ist für die Aufbereitung der Daten welche von der *View* Schicht dargestellt wird verantwortlich.

Die *Model* Schicht in der Abbildung 3.2 stellt die Daten Objekte für die Applikation zur Verfügung. Eine Darstellung der in der *RadioTour* App verwendeten Domain Objekte ist in der Abbildung 3.3 ersichtlich.

Die *Util* Schicht in der Abbildung 3.2 beinhaltet die Dienstobjekte für den Datenbank Zugriff, sowie für die Kommunikation der Applikation mit dem cnlab Server. Im Paket *utils* befinden sich allgemein gebrauchte Hilfsmethoden.

3.3 Klassendiagramm

Die Domainlogik beinhaltet die Kernelemente der Applikation. Einerseits sind dies die Rennfahrer, welche Informationen über sich festhalten, andererseits die Etappe mit den Informationen zur Strecke. Während dem Rennen werden die Fahrer in Gruppen unterteilt. Auch diese Gruppen sind in der Domain abgebildet. Das Klassendiagramm des Domain Package (siehe Abbildung 3.3) zeigt die wesentlichen Elemente.

Die Klasse *Rider* speichert die Angaben zu einem Fahrer und beinhaltet keine eigene Logik. Objekte dieser Klasse dienen als Stammdaten für alle Etappen welche in der Applikation erfasst sind.
Um die Fahrer nach Team sortiert anzeigen zu können, wird die Klasse *Team* genutzt.
In *Stage* ist die Etappe definiert. Jede Etappe hat eine Marschtabelle in Form von mehreren *PointOfRace* Objekten. Diese Objekte werden durch den Import der Marschtabelle erstellt.
Da pro Etappe jeder *Rider* einen anderen *RiderState* haben kann, gibt es die Verbindungsklasse *RiderStageConnection*. In dieser Klasse ist jeweils die Etappe mit dem Fahrer verknüpft. Dies ermöglicht es, den Rückstand eines Fahrers in mehreren Etappen differenziert zu verfolgen. Dadurch werden bei einem Etappenwechsel innerhalb der Applikation immer die Informationen zur aktuelle ausgewählten Etappe verwendet.
Die Klasse *SpecialRanking* stellt ein Spezialklassement dar und wird benötigt, um

2. Android Developers Reference, `http://developer.android.com/guide/topics/fundamentals/fragments.html`, aufgerufen am 31.05.2012.

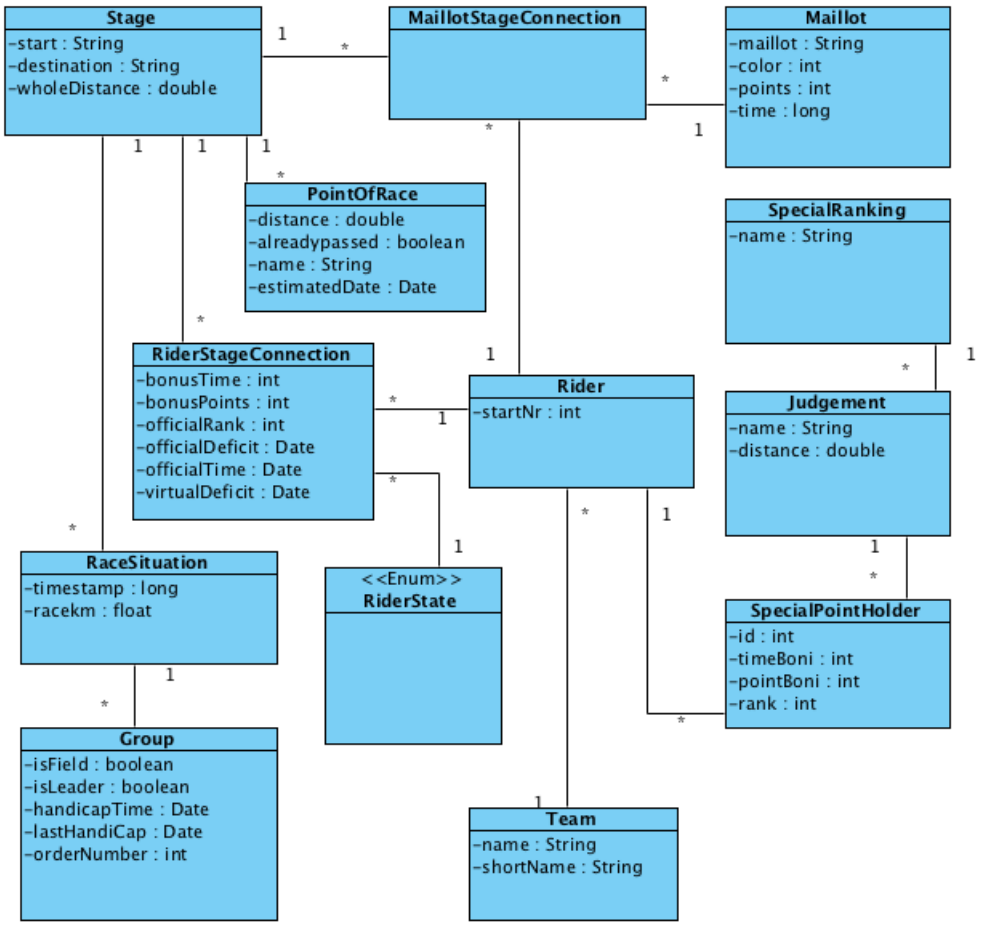


Abbildung 3.3: Die Domainklassen in der Abhängigkeit

die einzelnen Wertungen (*Judgement*), welche Etappen basiert sind, auch über alle Etappen hinweg zu verrechnen. Damit können zum einen Punkte- bzw. Zeitboni pro Etappe verrechnet werden, was zur Darstellung im Virtuellen Klassement gebraucht wird und zum andern eine Rangliste pro Spezialklassement geführt werden. Diese Rangliste enthält die Ergebnisse aller Wertungen des angegebenen Spezialklassements ohne Rücksicht auf die zugehörige Etappe.
Die *SpecialPointHolder* Klasse dient dazu, den zu Wertungen zugewiesenen Fahrern in der Datenbank zu speichern und zur späteren Berechnung des virtuellen Klassements zu berücksichtigen.
Eine *RaceSituation* stellt einen Stand von *Group* Objekten zu einem speziellen Zeitpunkt und Kilometerstand dar. Objekte vom Typ *RaceSituation* werden bei jeder Neugruppierung von Fahrern oder Setzen eines Rückstandes einer Gruppe erstellt, an den Server gesendet und in die lokale SQLite Datenbank geschrieben. Jeweils beim Start der Applikation sowie beim Wechseln der aktuellen Etappe wird die zuletzt in der Datenbank abgespeicherte *RaceSituation* geladen. Die *Group* Klasse enthält die Angaben zum aktuellen Rückstand zur Spitze, dem letzten bekannten Rückstand sowie boolsche Attribute welche bestimmen ob die Gruppe das Feld oder die Spitze darstellt.
Mit Hilfe der *Maillot* Klasse können Maillots erstellt und verteilt werden, welche dann über alle Etappen bestehen. Da nach jeder Etappe ein anderer Fahrer ein Maillot tragen kann, wird eine Verbindung zwischen der Etappe und einem Maillot gebraucht. Diese Funktion wird durch die Klasse *MaillotStageConnection* erfüllt.

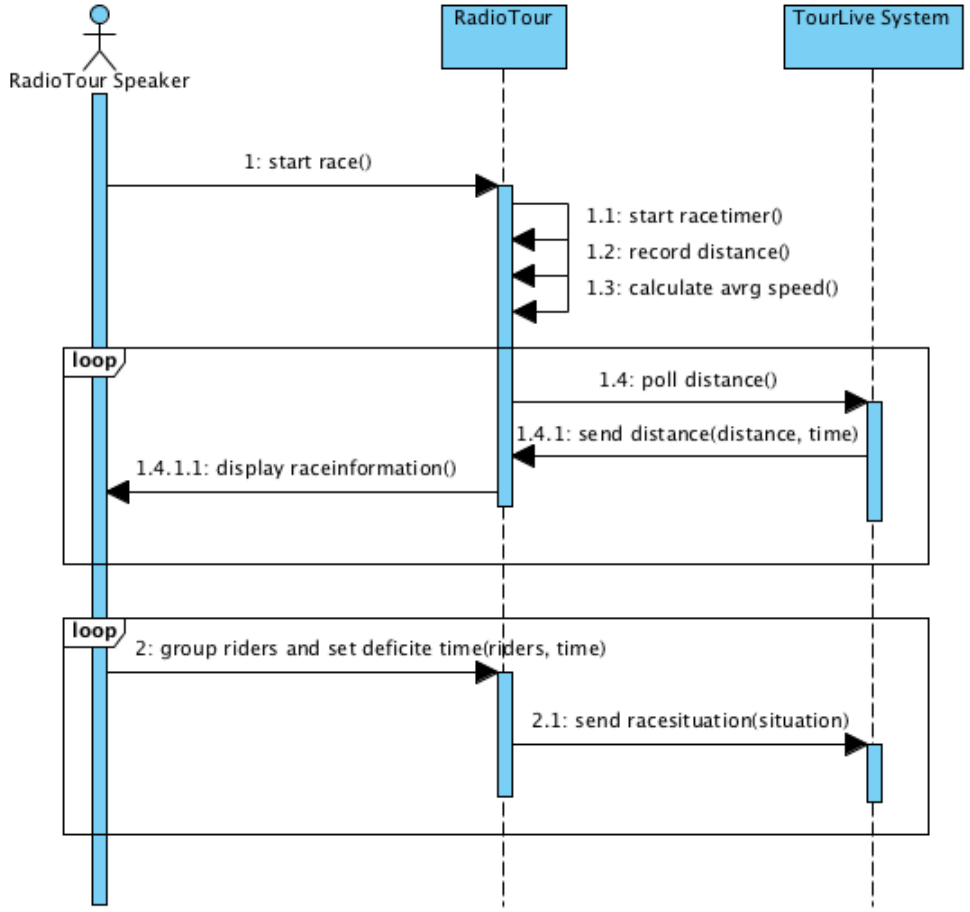
3.4 Datenbankschema

Die Nutzung von ORMLite zur Objektrelationalen Abbildung von Instanzen in die SQLite Datenbank impliziert, dass das Datenbankschema der Klassenstruktur, wie in Abbildung 3.3 dargestellt, entspricht.

3.5 Sequenz Diagramm

Der häufigste UseCase besteht darin, die Rennsituation zu erfassen und an den Server zu übermitteln. Gleichzeitig werden im Hintergrund die Live Informationen aktualisiert. Diese beiden Hauptanwendungsfälle sind im folgenden System Sequenz Diagramm dargestellt. Der Akteur wird durch den *RadioTour Speaker* dargestellt und das System durch die *RadioTour* Applikation. Das TourLive System stellt die Serverseite dar.

Abbildung 3.4: Das System Sequenz Diagramm



Das Rennen wird durch das Starten der Rennzeit gestartet. Ab diesem Zeitpunkt beginnt die Aufzeichnung des Rennkilometers und die Berechnung der durchschnittlichen Geschwindigkeit.

4 Realisierung

Dieses Kapitel widmet sich der Realisierung der Applikation inklusive der grafischen Erscheinung. Zudem werden detaillierte Informationen, um einem Entwickler beim Industriepartner einen schnellen Einstieg in das Weiterführen des Projektes zu ermöglichen, vermittelt.

4.1 Aktuelle Erscheinung

Activity

Die *RadioTour* App besteht aus einer Activity wie in Abbildung 4.1 dargestellt wird. Diese Activity wird in zwei Hauptteile eingeteilt. Es sind dies der (1) Header- sowie der (2) Hauptbereich.

Activity:

```
package ch.hsr.sa.radiotour.activities;
public class RadioTourActivity extends Activity
        implements Observer, OnClickListener
```

View definiert in: *res/layout/base_activity.xml*

Die *RadioTourActivity* Klasse ist der Startpunkt der Applikationsausführung und deshalb im *AndroidManifest.xml* File als Startactivity eingetragen. Diese Datei muss in jedem Android Projekt im Stammverzeichnis vorhanden sein. Es enthält wichtige Informationen zur Applikation, unter anderem die Startactivity.

4.1.1 Headerbereich

```
package ch.hsr.sa.radiotour.fragments;
public class HeaderFragment extends Fragment
        implements Observer, TimePickerIF
```

View definiert in: *res/layout/header_fragment.xml*

Das *HeaderFragment*, welches in Abbildung 4.2 illustriert ist, enthält alle Informationen, welche zu jedem Zeitpunkt der Applikationsausführung sichtbar sind. Dazu gehört der Systemzustand der zusätzlichen TourLive Komponenten, welche via den TourLive Server über JSON die *RadioTour* Applikation mit Informationen versorgt.

Zudem werden wichtige Angaben zum derzeitigen Stand des Rennens im *HeaderFragment* angezeigt, welche direkt aus der Tablet Anwendung stammen. Es sind dies die aktuelle Rennzeit und der aktuelle Rennkilometer, welcher, wie auch die aktuelle Höhe über Meer und die durchschnittliche Geschwindigkeit, aus dem im Tablet integrierten GPS Empfänger stammen. Um Fehlangaben zu korrigieren, ist die

Abbildung 4.1: RadioTour Activity

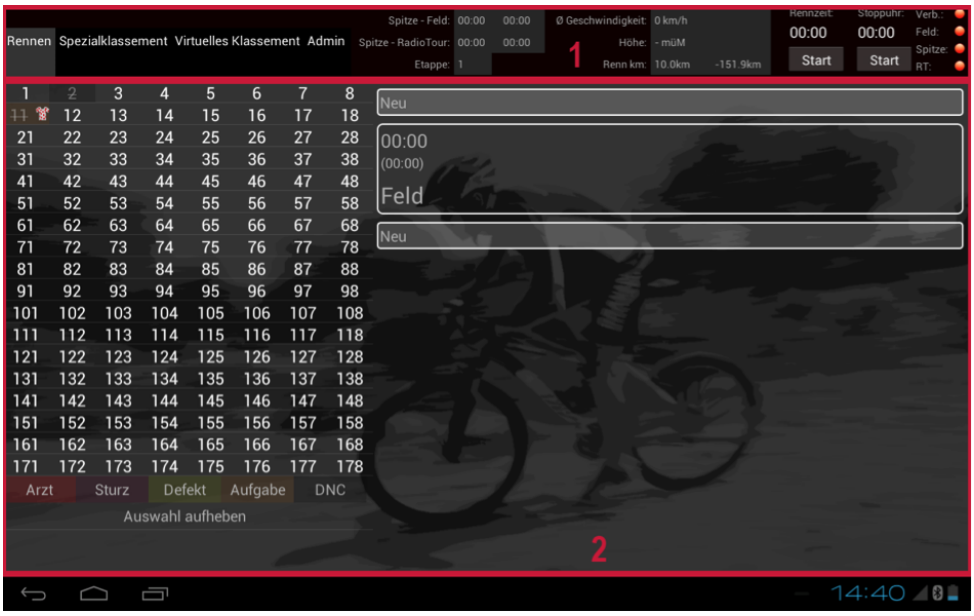


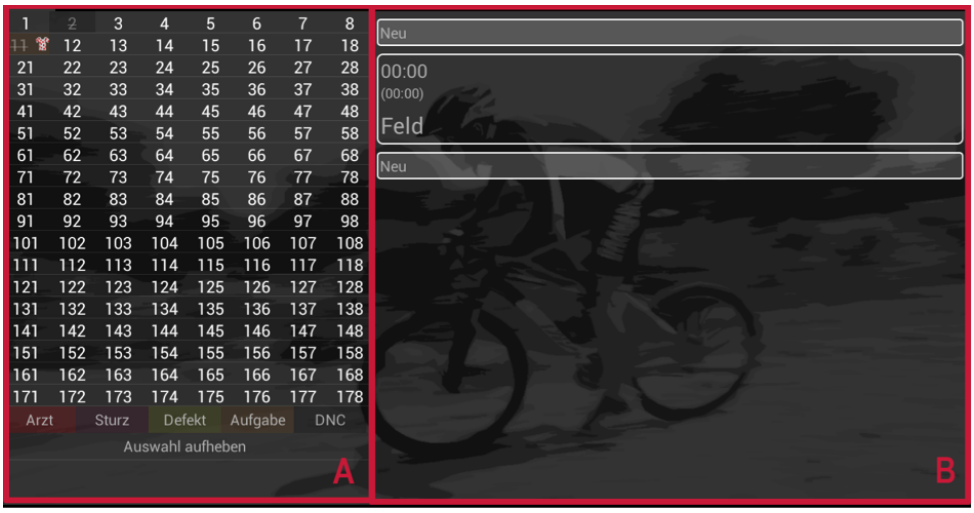
Abbildung 4.2: HeaderFragment

Kilometer- sowie die Rennzeitangabe editierbar. Zeitabstände werden mit der ebenfalls im *HeaderFragment* vorhandenen Stoppuhr gemessen. Mit einem Klick auf den „Etappe“ Text wird die Marschtabelle, welche bereits absolvierte Stationen ausgraut, angezeigt.

4.1.2 Hauptbereich

Der Hauptbereich wird je nach ausgewähltem Tab (im Headerbereich) mit einem anderen Fragment ausgefüllt.

Abbildung 4.3: RaceFragment



RaceFragment.java

```
package ch.hsr.sa.radiotour.fragments;
public class RaceFragment extends Fragment
```

View definiert in: *res/layout/race_layout.xml*

Das *RaceFragment* welches in Abbildung 4.3 zu sehen ist, beinhaltet zwei weitere Fragments. Das *RaceFragment* an sich beinhaltet keine Funktionalität und dient nur zum Zusammenführen der zwei Fragmente *DriverPickerFragment* und *RiderGroupFragment*

RiderPickerFragment

```
package ch.hsr.sa.radiotour.fragments;

public class RiderPickerFragment extends ListFragment
        implements OnClickListener
```

Das *RiderPickerFragment*, Fragment A in Abbildung 4.3 ermöglicht die Auswahl von einem oder mehreren Fahrern für die Zuweisung in eine Gruppe oder einem Spezialevent[1]. Die ausgewählten Fahrer können entweder per Drag and Drop oder mit anklicken des Zielfeldes zugewiesen werden. Mit einem Klick auf den „Auswahl aufheben“ Text werden die bereits ausgewählten Fahrer wieder deselektiert.

Die Darstellung der Daten wird mithilfe eines *ArrayAdapter<Team>* erzeugt. Dabei wird für jedes Team Objekt eine Zeile mit den jeweiligen Fahrern generiert. Die Textfelder mit den Spezialevents werden über eine dem *RiderPickerFragment* hinzugefügten *FooterView* ergänzt.

RiderGroupFragment

```
package ch.hsr.sa.radiotour.fragments;

public class RiderPickerFragment extends Fragment
```

View definiert in: *res/layout/group_fragment.xml*

Das *RiderGroupFragment*, Fragment B in Abbildung 4.3, stellt die aktuelle Rennsituation dar. Die schmaleren grauen Feldern dienen dazu, neue Gruppen zu erstellen. Wenn im *RiderPickerFragment* Fahrer ausgewählt sind, können diese in die graue Fläche gezogen werden (Drag and Drop) oder auf die Fläche geklickt werden. Den im transparenten Feld gruppierten Fahrer kann ein Rückstand relativ zur Spitze eingetragen werden. Dieser Rückstand wird den Fahrern welche dieser Gruppe angehören berechnet und im virtuellen Klassement berücksichtigt.

SpecialRankingFragment

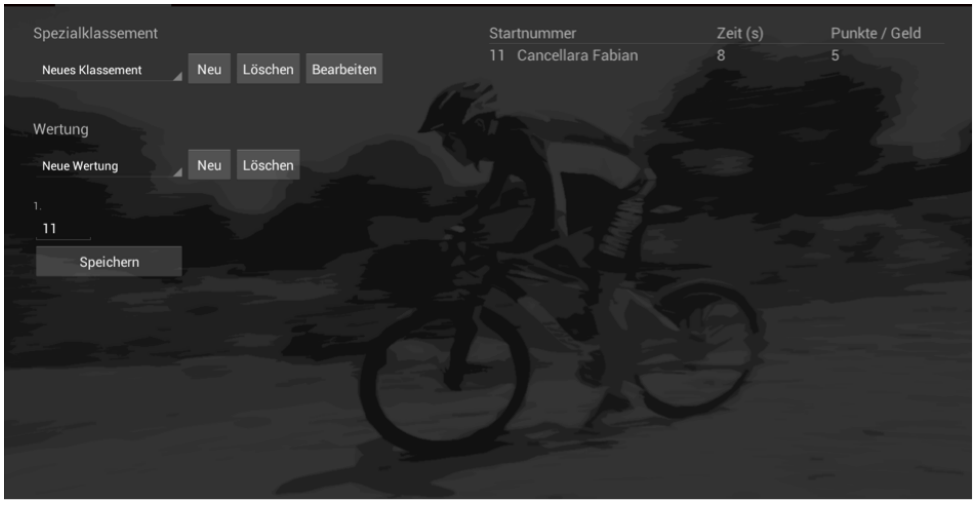


Abbildung 4.4: SpecialRanking Fragment

1. siehe Abschnitt 2.1

```
package ch.hsr.sa.radiotour.fragments;

public class SpecialRankingFragment extends Fragment
```

View definiert in: *res/layout/special_ranking_fragment.xml*

Das *SpecialRankingFragment*, abgebildet in Abbildung 4.4, erlaubt es neue Spezialklassemente und deren Wertungen, welche einer Etappe zugeordnet werden, zu erstellen. Eine Wertung beinhaltet Zeit- und/oder Punktebonus. Zudem kann jede Wertung eine unterschiedliche Anzahl berücksichtigte Fahrer haben. Die Boni, welche Fahrer die in einer Wertung eine Klassierung erreichen, werden pro Spezialklassement rechts im *SpecialRankingFragment* angezeigt. Die etappenbasierte Verrechnung der Boni geschieht im *VirtualRankingFragment*.

VirtualRankingFragment

Abbildung 4.5: VirtualRanking Fragment

Position	Start-Nr.	Name	Team	Land	Rang	Punktebonus	Virt. Rückstand	Offizielle Zeit	Offizieller Rückstand
1	1	Albasini Michael	OGE	SUI	0	0	00:00	00:00	00:00
2	2	O'Grady Stuart	OGE	AUS	0	0	00:00	00:00	00:00
3	3	Davis Allan	OGE	AUS	0	0	00:00	00:00	00:00
4	4	Cooke Baden	OGE	AUS	0	0	00:00	00:00	00:00
5	5	Meyer Cameron	OGE	AUS	0	0	00:00	00:00	00:00
6	6	Langeveld Sebastian	OGE	NED	0	0	00:00	00:00	00:00
7	7	Docker Mitchell	OGE	AUS	0	0	00:00	00:00	00:00
8	8	Wilson Matt	OGE	AUS	0	0	00:00	00:00	00:00
9	11	Cancellara Fabian	SCN	SUI	0	5	167:59:52	00:00	00:00
10	12	Freiburghaus Sepp	SCN	SUI	0	0	00:00	00:00	00:00
11	13	Jost Kevin	SCN	SUI	0	0	00:00	00:00	00:00
12	14	Kohler Martin	SCN	SUI	0	0	00:00	00:00	00:00
13	15	Reichenbach S_bastien	SCN	SUI	0	0	00:00	00:00	00:00
14	16	Saggiorato Mirco	SCN	SUI	0	0	00:00	00:00	00:00
15	17	Vogel Florian	SCN	SUI	0	0	00:00	00:00	00:00
16	18	Zahner Simon	SCN	SUI	0	0	00:00	00:00	00:00
17	21	Elmiger Martin	ALM	SUI	0	0	00:00	00:00	00:00
18	22	Mondory Lloyd	ALM	FRA	0	0	00:00	00:00	00:00
19	23	Roche Nicolas	ALM	IRL	0	0	00:00	00:00	00:00
20	24	Berard Julien	ALM	FRA	0	0	00:00	00:00	00:00

```
package ch.hsr.sa.radiotour.fragments;

public class VirtualRankingFragment extends ListFragment
```

Das *VirtualRankingFragment* erbt von ListFragment und zeigt die zu der aktuell ausgewählten Etappe die dazugehörigen *RiderStageConnection* (siehe Abbildung 3.3) an. Mit einem Klick auf einen Fahrer in der Rangliste lassen sich die Stammdaten des jeweiligen Fahrers verändern. So können zum Beispiel versehentlich als ausgeschieden markierte Fahrer wieder als „im Rennen“ gesetzt werden. Über Klicks auf die Spaltenbezeichnung lässt sich die Liste, sowohl auf- wie auch absteigend, nach den einzelnen Werten sortieren. Die Bezeichnung der aktuell sortierten Spalte wird fett dargestellt. Die Sortierung basiert auf der abstrakten *RiderSortStrategy* Klasse, welche nach dem Vorbild des *Strategy Pattern* implementiert wurde.

AdminFragment

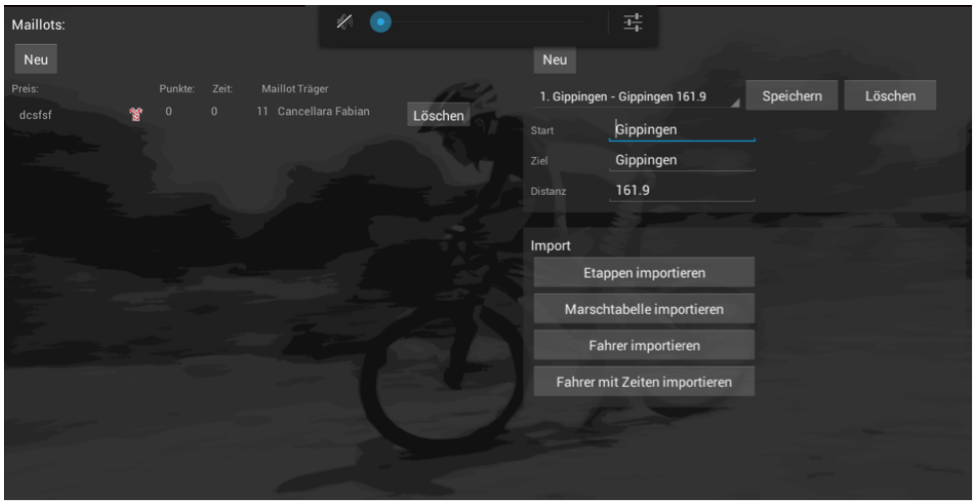
```
package ch.hsr.sa.radiotour.fragments;

public class AdminFragment extends Fragment
```

View definiert in: *res/layout/admin_fragment.xml*

Das *AdminFragment*, Abbildung 4.6, beinhaltet drei Grundfunktionen der Applikation. Zum einen ist dies die gesamte Maillotsverwaltung inklusive Zuweisung eines Maillots zu einem Fahrer, der in der aktuellen Etappe dieses Maillot trägt. Weiter wird im *AdminFragment* die aktuelle Etappe ausgewählt und es können neue Etappen erfasst oder bereits bestehende Etappen bearbeitet werden. Als letzte der drei

Abbildung 4.6: AdminFragment



Grundfunktionen steht der Import über Comma Separated Values (CSV) zur Verfügung. Über einen Dateibrowser welcher in die Applikation integriert ist können die CSV - Dateien ausgewählt und importiert werden.

4.2 Kommunikation zum Server

Es werden zwei Verbindung mit dem TourLive Server aufgebaut. Zum einen werden die Live Informationen aus dem TourLive Aufnahmesystem bei bestehender Verbindung alle 10 Sekunden abgefragt. Zum anderen wird die aktuelle Rennsituation an den Server übermittelt. Um einem Unterbruch der Verbindung möglichst unbemerkt zu überbrücken kommen zwei verschiedene Verfahren zum Einsatz.

4.2.1 Sending

Die Rennsituation bildet sich aus den gruppierten Fahrern mit den eingetragenen Rückständen zusammen. Bei jeder Änderung der Situation wird der aktuelle Stand mit dem Rückstand der jeweiligen Gruppen in eine Warteschlange geschrieben. In einem Hintegrundprozess wird bei bestehender Verbindung mit dem Server diese Warteschlange abgearbeitet. Gesendet wir dann die Rennsituation im JSON Format, ein Beispiel davon ist in der Abbildung 4.7 zu sehen. Da der Hintergrundprozess mit einer niedrigen Priorität läuft, wird weder das UserInterface noch andere Funktionen des Tablets belastet. Dieses Verfahren erlaubt es auch bei längeren Unterbrechungen sämtliche Updates in der richtigen Reihenfolge an den Server zu übermitteln.

Abbildung 4.7: Rennsituation im JSON Format

```
occurs in
ch.hsr.sa.radiotour.technicalservices.connection. JsonSendingQueue.java

{
      "timestamp":1338284032240,
      "situation":
            [
                  {
                        "drivernumber":[11, 15, 31],
                        "isLeader":true,
                        "groupnr":0,
                        "handicaptime":-2209078800000,
                        "isField":false
                  },
                  {
                        "isLeader":false,
                        "groupnr":1,
                        "handicaptime":-2209078800000,
                        "isField":true
                  },
                  {
                        "drivernumber":[76, 18, 41, 35],
                        "isLeader":false,
                        "groupnr":2,
                        "handicaptime":0,
                        "isField":false
                  },
                  {
                        "drivernumber":[178, 103, 67],
                        "isLeader":false,
                        "groupnr":3,
                        "handicaptime":0,
                        "isField":false
                  }
            ]
}
```

4.2.2 Receiving

Im *HeaderFragment* werden die Abstände zwischen der Spitze und dem Feld sowie der Spitze und dem *RadioTour* Auto in Kilometer und Sekunden angegeben. Diese Informationen kommen aus dem TourLive Aufnahmesystem und werden auf dem Server berechnet. Dabei kommt ein Hintergrundprozess zum Einsatz, jedoch wird dafür ein Polling verwendet. Beim Polling wird in einem vordefinierten zeitlichen Abstand eine HTTP Anfrage an den Server geschickt. Kommt keine Antwort oder kann die Anfrage gar nicht erst platziert werden, konnte keine Verbindung aufgebaut werden. In diesem Fall werden die bestehenden Informationen weiter angezeigt, jedoch mit dem Hinweis, dass das Aufnahmesystem nicht verfügbar sei. Bei der Anfrage an den Server erwartet die Applikation eine Antwort im JSON Format wie in der Abbildung 4.8 dargestellt wird.

Abbildung 4.8: Livedaten im JSON Format

```
occurs in
ch.hsr.sa.radiotour.technicalservices.connection.LiveData.java

{
sources:
      [
            [
                  {
                        source: "Spitze",
                        timestamp: "15:27:52",
                        rennkilometer: "-1",
                        online: "true"
                  }
            ],
                  [
                        {
                        source: "RadioTour",
                        timestamp: "0",
                        rennkilometer: "-1",
                        online: "false"
                        }
                  ],
            [
                  {
                        source: "Feld",
                        timestamp: "0",
                        rennkilometer: "-1",
                        online: "false"
                  }
            ],
      ]
}
```

ORMLite (Version 4.39)

Zur Speicherung der Objekte in die SQLite Datenbank des Tablet wird die Java Library ORMLite benutzt. In der Zwischenzeit sind neue Releases der Library auf der Webseite `http://ormlite.com/releases/` erschienen. Für weitere Informationen zur Library wird auf `http://ormlite.com/sqlite_java_android_orm.shtml` verwiesen.
Um eine Änderung an den zu persistierenden Objekten durchzuführen ist es notwendig, die Klasse DatabaseConfigUtil.java im

```
package ch.hsr.sa.radiotour.technicalservices.database;
```

als normale Java Applikation auszuführen. Dies generiert das *Configuration File*

res/raw/ormlite_config.txt

welches für eine effizientere Ausführung von *ORMLite* gebraucht wird.

Um eine Klasse für die Persistierung durch *ORMLite* vorzubereiten, muss diese mit Java Annotationen versehen werden. Als Beispiel für das Vorgehen wird der relevante Teil der annotierten Klasse *MaillotStageConnection* im folgenden Listing gezeigt.

```
@DatabaseTable
public class MaillotStageConnection {
        @DatabaseField(generatedId = true)
        int id;
        @DatabaseField(foreignAutoRefresh = true,
        foreign = true, columnName = "maillot")
        private Maillot maillot;
        @DatabaseField(foreignAutoRefresh = true,
        foreign = true, columnName = "etappe")
        private Stage stage;
        @DatabaseField(foreignAutoRefresh = true,
        foreign = true, columnName = "rider")
        private Rider rider;
```

5 Testing

5.1 User Interface Tests

Die Domain Klassen der *RadioTour* Applikation enthalten wenig bis gar keine Logik. Aus diesem Grund wird auf herkömmliche Java Unit Test (JUnit) verzichtet. Als Ersatz dazu werden zwei Ansätze zum Testen des User Interfaces verwendet. Zum einen ist ein Testprojekt mit Hilfe des Test Frameworks Robotium vorhanden. Andererseits wird der Android SDK interne Monkey verwendet.

5.1.1 Robotium Testprojekt

Mit Hilfe von Robotium werden Klicks sowie Texteingaben an die Applikation gesendet. Das Testprojekt ist deshalb eine Form des automated User Interface Testings. Wird das Testprojekt gestartet führt es folgende Aufgaben durch:

- Starten der Applikation
  Das Testprojekt startet die Applikation und wartet das erfolgreiche Ausblenden des Splashscreen ab.
- Importieren von Fahrern
  Falls die Applikation keine Fahrerinformationen enthält, navigiert das Testprojekt mit Hilfe von gesendeten Klicks zum Admin Bereich der Applikation und wählt das richtige Import CSV-File aus.
- Gruppieren von Fahrern
  Das Testprojekt wählt zwei Fahrer aus und setzt diese an die Spitze. Dabei wird getestet ob nach dem Auswählen wirklich zwei Fahrer ausgewählt sind und nach dem Gruppieren wieder keine.
- Testen der Gruppen Konsistenz
  Es wird getestet, ob jeweils nur eine Gruppe mit der Bezeichnung „Spitze“ sowie nur eine mit der Bezeichnung „Feld“ vorhanden ist.
- Neustart der Applikation
  Das Testprojekt schliesst die Applikation und startet sie neu. Dabei wird überprüft, ob der Gruppenstand vor dem Schliessen der Applikation derjenigen nach dem Neustart entspricht.
- Neues Maillot erstellen
  Mit Hilfe der gesendeten Klicks wird ein neues Maillot erstellt und ein Fahrer als Träger dieses Maillots definiert. Darauf wird überprüft, ob zum angegeben Fahrer der Eintrag zum Maillot vorhanden ist.

- Neues Spezialklassement erstellen
  Das Testprojekt navigiert zum Bereich Spezialklassement und erstellt dort ein neues Spezialklassement sowie eine neue Wertung. Zu dieser Wertung werden die Gewinner eingegeben. Darauf wird überprüft, ob die Bonuspunkte, sowie die Bonuszeit, bei den angegebenen Fahrern korrekt eingetragen ist.

- Löschen der oben erstellten Objekte
  Die in den oben aufgeführten Aufgaben erstellten Objekte werden - bis auf die Fahrer - gelöscht. Nach dem Löschen überprüft das Testprojekt ob sich alle Fahrer wieder in der Ausgangslage befinden.

5.1.2 Android Monkey

Der Android Monkey ist ein Kommandozeilentool welches auf jedem Android Gerät sowie dem Emulator ausgeführt werden kann. Das Tool sendet pseudo-zufällige Benutzer Events in schnellstmöglicher Abfolge an das System welches getestet wird. Dadurch werden Fehler im User Interface unabhängig von der darunter liegenden Logik aufgedeckt. Zum Zeitpunkt der Codeübergabe beendeten beide Testläufe fehlerfrei.

5.2 Feldtest

Um die Benutzerfreundlichkeit und den Mehrwert der Applikation im Vergleich zur bisherigen Web Applikation zu ermitteln, ist ein Feldtest unabdingbar. Bei der Berner Rundfahrt[1] bot sich der *RadioTour Speaker* David Loosli[2] an, die Applikation zu testen. In einem ersten Treffen wurden die grundlegenden Funktionen der Applikation und die Bedienung erklärt. Weiter wurde ein Ausschnitt aus einem fiktiven Rennen durchgespielt, wobei eine Person den Chronofunk simulierte. Ein kurzer Auszug aus der vorbereiteten Situation ist unten aufgeführt.

- Alle Fahrer sind importiert und das Rennen beginnt jetzt
- Die Rennzeit wird gestartet
- *Chronofunk:* Fahrer 4 & 17 von Beginn an, an der Spitze
- *Chronofunk:* Bereits 1:07 Vorsprung
- *Chronofunk:* 31 hat ein defektes Rad und muss raus
- *Chronofunk:* 8, 83 & 34 fallen hinter das Feld mit einem Rückstand von 4:31

Der vollständige Usability Test sowie der Testlauf an der *Berner Rundfahrt* mit den Rückmeldungen von Herrn Loosli sind im Anhang 7.5 zu finden.

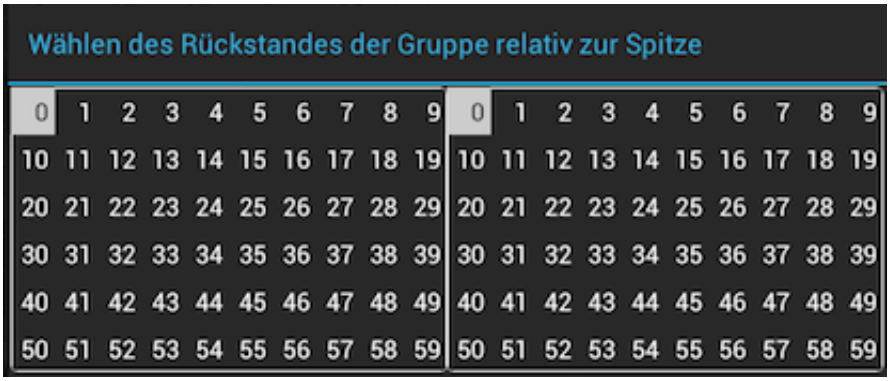
Mit den Erfahrungen und den Rückmeldungen aus diesem Event konnte das Endprodukt deutlich verbessert werden. Die wesentlichen Punkte, welche für die Weiterentwicklung verwendet wurden sind folgende:

- TimePicker Nummern (Auswahl des Rückstandes einer Gruppe) sind zu klein (Korrektur siehe Abbildung 5.1)

1. Berner Rundfahrt, `http://www.berner-rundfahrt.ch`, Aufgerufen am 01.05.2012.
2. David Loosli, ehemaliger profi Radrennfahrer und *RadioTour Speaker* der Berner Rundfahrt

- Nur die Fahrer, welche aufgegeben haben oder nicht erschienen sind, sollen ausgegraut werden (Vorschlag in Abbildung 5.2)
- Ein Fahrer kann folgende Status haben:
  - Im Rennen / aktiv
  - Nicht gestartet
  - Ausgeschieden
- Rennzeit und Kilometer sind nach einem Absturz der Applikation noch verfügbar

Abbildung 5.1: TimePicker - Auswahl eines Zeitrückstandes



Besonders hervorzuheben ist an dieser Stelle, dass Herr Loosli, entgegen den Erwartungen, die Fahrer, welche in einer Gruppe eingeteilt sind (z.B. Spitze), im RiderPicker nicht ausgegraut haben möchte. So entstand der in Abbildung 5.2 dargestellte Entwurf.

Abbildung 5.2: RiderPicker - Auswahl von Fahrern

Die Abbildung 5.2 zeigt eine Situation, bei der die Fahrer 33 und 106 nicht gestartet sind und die Fahrer 152 und 154 aufgegeben haben. Diese Fahrer sind farblich markiert und durchgestrichen. Die weiteren Farben entsprechen den besonderen Ereignissen: Arzt, Sturz oder Defekt eines Fahrers bzw. eines Fahrrads.

6 Ergebnisse und Schlussfolgerungen

In diesem Kapitel werden die erreichten Ziele und das Endprodukt zusammengefasst und im Rahmen der Aufgabenstellung beurteilt. Des weiteren wird die Zukunft des Produktes und die weitere Entwicklung erörtert.

6.1 Endprodukt

Das Ziel war es, eine verbesserte und modernisierte Applikation für den *RadioTour Speaker* zu erstellen, welche die bisherige Web Applikation ersetzt. Die Anforderungen bestanden aus den Features der bestehenden Lösung und wurden weitgehend erreicht. Zudem wurden die Wünsche und Anregungen des *RadioTour Speakers* implementiert. Das Endprodukt beinhaltet die Android Applikation *RadioTour* mit der passenden Hardware, einem Samsung Galaxy Tab 10.1. Diese Lösung ist im heutigen Stand, für die Begleitung von Radrennen, einsatzfähig. Dies konnte durch den Einsatz an der Berner Rundfahrt bestätigt werden. Für die Entwicklung von weiteren Features und der Vorbereitung für die Verwendung an der Tour de Suisse wird die Applikation dem Industriepartner, der cnlab AG, übergeben.

Gegen Ende des Projekts kamen noch weitere Anforderungen hinzu. So z.B. der Import der Fahrer direkt vom Server via JSON oder das automatische Navigieren zur aktuellen Rennposition in der Marschtabelle. Da diese Anforderungen erst bei einem fortgeschrittenen Stadium des Projekts hinzukamen, konnten diese Features nicht mehr implementiert werden. Es handelt sich bei diesen Anforderungen nicht um kritische Bereiche, daher sind die Änderungen auch in Form eines Updates noch möglich.

6.2 Ausblick

Der nächste und letzte Feldtest der Applikation vor der Tour de Suisse erfolgt an den Radsporttage in Gippingen[1]. Da dieser Feldtest erst nach der Projektabgabe stattfindet, liegt er in der Obhut der cnlab AG. Das Feedback aus diesem Test wird zeigen, wie gut die Rückmeldungen aus der Berner Rundfahrt umgesetzt werden konnten.

Bis zur Tour de Suisse gibt es also noch die Möglichkeit, Anpassungen vorzuneh-

1. Radsporttage Gippingen, `http://www.gippingen.ch/`, Aufgerufen am 30.05.2012.

men. Nach der Tour de Suisse, dem ersten Einsatz in einem Etappenrennen, ist eine weitere Auswertung und allfällige Optimierung sinnvoll. Mit dieser Arbeit wurde ein wichtiger Grundstein gelegt, mit dem weiter gearbeitet werden kann.

Abbildungsverzeichnis

Glossar

API
Application Programming Interface. 19, 24

Chronofunk
Die Motorradfahrer, welche im Rennfeld verteilt mitfahren und die Positionen der Ausreissergruppen per Funk an den RadioTour Speaker übermitteln. 38

CSV
Ein Dateiformat, bei welchem die Datensätze über ein Trennzeichen voneinander getrennt sind.. 33, 37, 56

HTTP
Das Hypertext Transfer Protocol (HTTP) ist ein weit verbreitetes Protokoll, um Daten und Inhalte im Web zu übertragen. In dieser Arbeit wird es in der Kommunikation mit dem Server verwendet.. 15, 34

IDE
Integrated Development Environment. 3, 18, 20

JSON
JavaScript Object Notation (JSON), ist eine Notation zur Darstellung von Objekten in Textform. 3, 29, 33, 34, 41

JUnit Test
Java bietet die Möglichkeit integrierte Softwaretests automatisiert durchzuführen. Dies erleichtert die Arbeit enorm und unterstützt ein Entwicklungsteam, eine möglichst hohe Testabdeckung zu erarbeiten. 37

ORM
Object-relational mapping. 19

RUP
Rational Unified Prozess. 10

Splashscreen
Eine Anzeige, die oftmals beim Start einer Applikation die Wartezeit bis zur vollständigen Initialisierung überbrückt.. 18, 37

SQLite
SQLite ist eine Cross Plattform Datenbankengine, welche ohne Konfiguration auskommt. Es handelt sich dabei um eine Datenbank in einer Datei. 3, 15, 19, 26, 27, 36

Literaturverzeichnis

[1] Robbie Matthews. *Beginning Android Tablet Programming (Beginning Apress)*. Apress, 1 edition, 2011.

[2] Donn Felker. *Android Tablet Application Development For Dummies*. For Dummies, 1 edition, 2011.

[3] Sven Riedel. *Git- kurz & gut*. O'Reilly, 1 edition, 2009.

[4] Thomas Künneth. *Android 3: Apps entwickeln mit dem Android SDK (Galileo Computing)*. Galileo Computing, 1 edition, 2011.

[5] cnlab AG, Dr. Prof. Peter Heinzmann. Administratives zu studien- und bachelorarbeiten. `http://www.cnlab.ch/kurse/SABA/`, 2012. letzter Zugriff am 30.05.2012.

7 Anhang

7.1 Projektmanagement

Für die Projektorganisation kommen verschiedene Hilfsmittel zur Anwendung. Im Folgenden werden die einzelnen Bereiche behandelt.

7.1.1 Zeitplan

Die Zeiterfassung ist in einem dafür vorbereiteten Excel File abgelegt. Folgende Meilensteine sind im Zeitplan festgelegt:

Projektstart		21.02.2012
Milestone 1	*RadioTour* Bestandesaufname Webanwendung	05.03.2012
Milestone 2	Plattform Entscheid gefällt	12.03.2012
Milestone 3	Features und Abgrenzung definiert und mit Heinzmann besprochen	19.03.2012
Milestone 4	Prototyp “Rennsituation„ bei Heinzmann zum Testen übergeben	09.04.2012
Milestone 5	Dokumentation Teil Einleitung zur Review übergeben	23.04.2012
Milestone 5.1	Prototyp zum Testen an Heinzmann übergeben	01.05.2012
Milestone 6	Feature Freeze	07.05.2012
Rennen	Berner Rundfahrt, Lyss	12.05.2012
Milestone 7	Code Freeze	14.05.2012
Milestone 8	Dokumentation Freeze	21.05.2012
Milestone 9	Abgabe komplett	28.05.2012
Projektende		01.06.2012
Rennen	Gippingen (nach Abgabe)	07.06.2012
Rennen	Tour de Suisse (nach Abgabe)	09.-16.06.2012

Die Auswertung der Zeiterfassung zeigt deutlich mehr (Ist-) Arbeitsstunden an, als die in der Planung eingetragenen (Soll-) Stunden. Besonders bei der Implementierung ist oftmals mehr Zeit benötigt worden als ursprünglich geplant war. Weiter hat das Erstellen der Dokumentation mehr Zeit in Anspruch genommen als erwartet. Der Projektzeitplan mit den vollständigen eingetragenen Arbeitsstunden ist in digitaler Form auf der beigelegten CD.

7.1.2 Code Base und Issue Tracking

Das Code Repository wurde auf Github[1] erstellt und verwaltet. Dieses Repository ist öffentlich lesbar, jedoch kann nicht öffentlich darauf geschrieben werden.
Nach der ersten Implementierungsphase sind die Probleme, Fehler und weiteren Features der Applikation im integrierten Issuetracker von Github erfasst. Die Issues sind einer der folgenden Kategorie zugeordnet:

- must
- can
- nice

Sämtliche „must“ und „can“ Issues wurden erfolgreich abgearbeitet. Offen geblieben ist ein einziges „nice“ Issue. Dabei geht es um die Auswahl von Fahrern für die Zuweisung von Wertungen in einem Spezialklassement.

7.2 Persönliche Berichte

7.2.1 Daniel Stucki

Am Anfang des Projekts herrschte für mich grosse Unklarheit und Skepsis gegenüber der Studienarbeit. Unter Schlagworten wie „RadioTour“, „Tour de Suisse“, „Radfahrer“ in der Ausschreibung des Projektes konnte ich mir nichts vorstellen oder die Begeisterung dafür hielt sich in Grenzen. Trotzdem konnte ich mich für dieses Projekt entscheiden und die Bewerbung zusammen mit Florian Bentele abschicken. Dies vor allem wegen der Tablet Erwähnung in der Beschreibung. Nach der ersten Sitzung mit dem betreuenden Dozenten und dem langsamen Durchschimmern der Tatsache, dass die Applikation auf Android entwickelt wird, war ich dann doch erleichtert. Im Herbstsemester 2011/2012 habe ich ihm Rahmen des Challenge Projekt Moduls bereits einige Erfahrungen im Bereich Android sammeln können. Ausserdem spielte mir meine Teilzeitbeschäftigung als Entwickler im Java Bereich erfreulich in die Karten.
Als Erstes stand die Analyse der alten webbasierten Applikation an, was für mich nicht wirklich spannend war, jedoch auch gemacht werden musste. Ich hätte mich am liebsten von Anfang an in die Programmierung gestürzt. Doch Florian Bentele hielt mich, im Nachhinein gesagt zum Glück, davon ab und bestand auf eine gründliche Planung des Vorgehens um unliebsame Überraschungen zu vermeiden. Nach einigen Wochen in der Analyse und Planung des Vorgehens konnte auch endlich die Implementierung starten. Schnell stellten sich erste Erfolge und Resultate ein und die Begeisterung fürs Projekt steigerte sich stetig. Drei Wochen vor Ende des Projektes hatten wir die Möglichkeit mit Herrn David Loosli, einem ehemaligen Profi Radfahrer, einen Usability Test durchzuführen. Am Tag darauf konnte ich an der Berner Rundfahrt mit dem RadioTour Speaker im Auto mitfahren und unsere Applikation einem 4.5 Stunden dauernden Feldtest unterziehen. Resultat: Kein Systemabsturz und ich konnte den erfahrenen RadioTour Speaker Chef, welcher die Aufgabe mit Block und Stift ausführte, mehrfach verbessern - was für ein Erfolg. Dieses positive Zwischenfazit motivierte uns insgeheim und spornte uns noch mehr an. In den letzen zwei Wochen standen noch Feature Implementierungen an, welche erst dann als

1. Github, `https://github.com/dstucki/RadioTour`, Aufgerufen am 11.05.2012.

Requirements definiert wurden. Dies warf unseren Zeitplan zur Erstellung der Dokumentation ziemlich durcheinander.
Die Zusammenarbeit mit Florian Bentele hat mir überhaupt keine Probleme bereitet. Wir kennen uns schon seit Studienbeginn im Herbst 2009, weshalb unerwartete Streitereien innerhalb des Teams von Beginn an als unwahrscheinlich angesehen werden konnten. Ausserdem amtete er als Überwacher des Projektstatus, damit wichtige Deadlines eingehalten wurden.
Aus diesem Projekt nehme ich ein fundiertes Wissen im Bereich Android mit. Ausserdem bewies mir das Projekt, dass Projektmanagement und Planung genau so wichtig sind wie Programmierkenntnisse.

7.2.2 Florian Bentele

Zu Beginn des Projekts war es für mich schwierig, den Arbeitsaufwand und die Erwartungen an mich abzuschätzen. Die Aufgabenstellung als Anstoss zu verstehen und wird ständig weiter entwickelt und konkretisiert. Da wir aber schnell in die Thematik gefunden haben und unsere Ideen bei jeder Besprechung ein Stück weiter zu einem einsetzbaren Produkt führten, schwanden sämtliche Unsicherheiten. Besonders der Reiz, eine Applikation zu entwickeln, die den Anforderungen eines Live Sport Events entsprechen muss und in der Realität eingesetzt werden soll, führten zu einer hohen Einsatzbereitschaft und grosser Motivation. Ich hatte eine gewisse Verantwortung übernommen, die Applikation am Ende des Semesters funktionstüchtig abzuliefern. Weiter war es mir ein Anliegen, eine Arbeit im Bereich Mobile oder Web Applikationen zu erstellen. Mit diesem Projekt gelang es mir, einen tiefen Einblick in die Entwicklung von Android Applikationen zu erarbeiten.
Die ersten Wochen verliefen dennoch harzig, da die Analyse der bestehenden Applikation sehr aufwändig und zeitintensiv war. Sämtliche Tools und das Projektmanagement wurden eingerichtet, die Aufgaben aufgeteilt. Bei der Implementierungsphase ging es dann sehr gut voran. Wir haben das Aufgabenumfeld in Teilprobleme aufgeteilt und diese dann abgearbeitet. In weiteren Projekten würde ich mehr Wert auf diese Aufteilung legen. Nach der fundamentalen Architektur, die in meinen Augen ein Prozess ist, bei dem das ganze Team dabei sein sollte, können die Features gut aufgeteilt werden. Nach dem Hauptteil der Implementierung hatten wir die einmalige Gelegenheit, unsere Applikation an einem Radrennen zu testen. Dies ermöglichte uns, ein echtes, unverfälschtes Feedback zu erhalten. Aus den Rückmeldungen konnten wir die Applikation weiter verbessern und für die Tour de Suisse bereit machen. Im weiteren Verlauf versuchten wir, den bestehenden Code zu optimieren und verschönern (Refactoring). Da aber bis in der zweitletzten Woche des Projekts noch Features hinzukamen, war dies eher schwierig. Gegen Ende des Projekts kamen immer mehr Aufgaben im Bereich Dokumentation dazu. Aber auch diesen Teil konnten wir gut aufteilen.
Die Zusammenarbeit mit Daniel Stucki hat optimal funktioniert. Ich konnte viel von seiner Erfahrung als Java Entwickler profitieren und wir haben die Herausforderung gemeinsam gemeistert. Insbesondere in der Entwicklung ist es für mich wichtig, Teilprobleme und deren Lösungsansätze mit einer involvierten Person diskutieren zu können. Ich freue mich, auch in der Bachelorarbeit mit Daniel Stucki zusammen arbeiten zu können.
Aus diesem Projekt nehme ich diverse Erfahrungen mit, einerseits das Erlernte im Bereich der Android Programmierung, andererseits aus dem Projektmanagement und der Wichtigkeit einer möglichst genauen Zeitplanung. Auch die Meilensteine und der Feldtest sind gute Methoden, um den Projektverlauf möglichst zeitgerecht einzuhalten.

7.3 Kaufempfehlung

In diesem Abschnitt wird der cnlab AG ein Android Tablet empfohlen, welches für den Einsatz während der Tour de Suisse geeignet ist. Dabei sind vor allem die zwingenden (Killer) Kriterien massgebend.

Killer Kriterien

- Android: Die Applikation wird in Java für die Android Plattform entwickelt.
- USB Anschluss: Für den Import der Fahrerdatenbank ist eine USB Schnittstelle zwingend erforderlich.
- 3G / WWAN: Um während des Rennens mit dem Server zu kommunizieren wird eine Internetverbindung benötigt.
- GPS/AGPS
- Autospeisung Adapter möglich

Optionale Kriterien

- Grosser Bildschirm: Da die Software in einer Echtzeitumgebung eingesetzt wird, sind für die Bedienung andere Anforderungen relevant als z.B. im Büro. Ein grosser Bildschirm ermöglicht es das Userinterface für diese Anforderung zu optimieren.
- Akkulaufzeit: Die längste Etappe dauert etwa 6h, für das Tablet ist demnach eine Akkulaufzeit von min 6h gewünscht.

Empfehlung

Das optimale Gerät für den Einsatz an der Tour de Suisse ist das Lenovo ThinkPad. Einzig der Preis ist bedeutend höher als bei der Konkurrenz. Falls also die Differenz des Preises in Kauf genommen werden kann, wird die Anschaffung des ThinkPad empfohlen. In der beiliegenden CD befindet sich die detaillierte Evaluation als pdf.

7.4 UseCases der bisherigen Applikation

Im unten stehenden UseCase Diagramm (Abbildung 7.1) sind die primären UseCases aufgeführt. Nur der *RadioTour Speaker* erfasst Daten in dieser Applikation und ist daher der einzige Akteur. Das System wird durch die *RadioTour* Applikation abgebildet. Zur besseren Darstellung werden einzelne UseCases vereinfacht oder zusammengefasst.

- **Fahrer auswählen**
  Dem *RadioTour Speaker* muss es möglich sein, einen oder mehrere Fahrer schnell auszuwählen. Die Fahrer werden im Auswahldialog bevorzugt durch ihre Startnummern dargestellt. An der Tour de Suisse besteht ein Team – nach Aussage von P. Heinzmann – aus 8 Fahrern. Um eine möglichst gute Übersicht zu gewährleisten, werden die Fahrer jeweils zeilenweise in deren Teams gruppiert. Ausgewählte Fahrer werden farblich hervorgehoben. Die Nummern der Fahrer, welche bereits Gruppen zugewiesen wurden, werden in Klammern dargestellt. Die Nummern ausgeschiedener Fahrer werden gestrichen dargestellt.

Abbildung 7.1: UseCase Diagramm

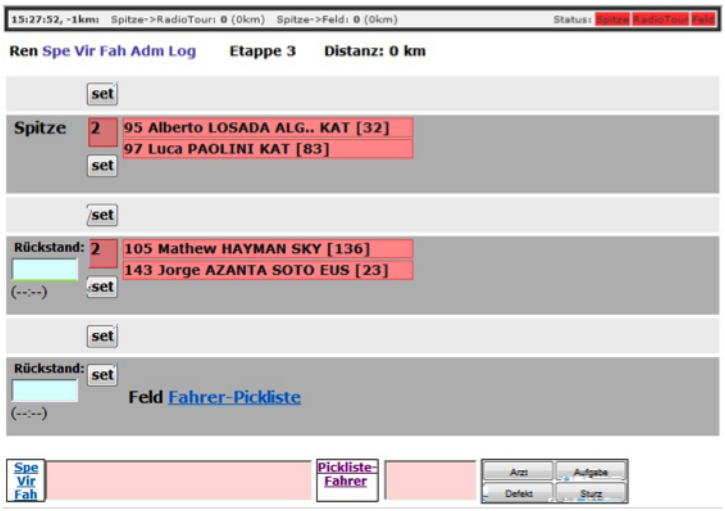
Abbildung 7.2: Die Fahrerauswahlliste zur Gruppierung in der bisherigen Web Applikation

set 3 Fahrer gewählt

1	2	3	4	5	6	7	8
11	12	13	14	15	16	17	18
21	22	23	24	25	26	27	28
31	~~32~~	33	34	35	36	37	38
41	42	43	44	45	46	47	~~48~~
51	~~52~~	53	54	55	~~56~~	57	58
61	62	63	64	65	~~66~~	67	68
71	~~72~~	73	74	75	76	77	~~78~~
81	82	83	84	85	~~86~~	~~87~~	88
~~91~~	92	93	94	(95)	96	(97)	98
101	102	103	104	(105)	106	107	108
111	112	113	114	115	116	~~117~~	118
121	122	~~123~~	124	125	126	127	~~128~~
~~131~~	132	133	134	135	136	~~137~~	138
141	142	(143)	144	145	146	147	~~148~~

- **Fahrer gruppieren**
  Dem *RadioTour Speaker* muss es möglich sein, die ausgewählten Fahrer in Gruppen zu organisieren. So kann er die ihm gemeldeten Rennsituationen mit Ausreissern, Verfolgern, Feld und Abgehängten darstellen.

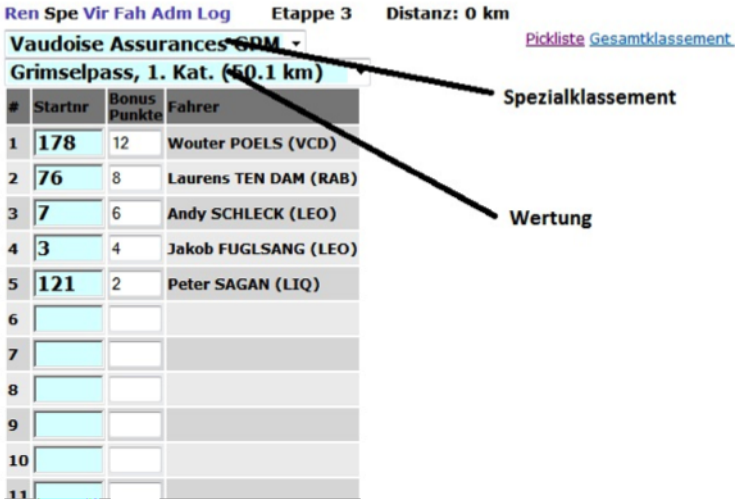
Abbildung 7.3: Gruppierung in der bisherigen Web Applikation



- **Rückstände angeben**
  Dem *RadioTour Speaker* muss es möglich sein, für die Gruppen (siehe oben) ihre jeweiligen Zeitabstände relativ zur Spitze einzugeben. Falls vorhanden, sollen auch die mit dem TourLive GPS-System erfassten Zeitabstände Spitze-Feld eingeblendet werden.

- **Spezial Events erfassen**
  Dem *RadioTour Speaker* muss es möglich sein, für ausgewählte Fahrer Spezialereignisse festzulegen. Dies sind beispielsweise Arztbesuch, Aufgabe, Defekt oder Sturz.

Abbildung 7.4: Die Spezialklassemente und Wertungen in der bisherigen Web Applikation



- **Spezialklassement und Wertung wählen**
  Bei Mehretappenrennen werden typisch neben dem Gesamtklassement (schnellster Fahrer) mehrere Spezialklassemente (z.B. Bergpreis, Sprint, Punkteklassement) gewertet. Innerhalb der Etappen gibt es jeweils mehrere Stellen (Wertungen), an denen für die Spezialklassemente Punkte vergeben werden. Dem *RadioTour Speaker* muss es möglich sein, die gewünschte Wertung zu einem der vorher erfassten Spezialklassemente auszuwählen.

- **Spezialklassement Resultate eintragen**
  Dem *RadioTour Speaker* muss es möglich sein, für eine Wertung, welche er ausgewählt hat (siehe UC oben), die Ränge zur Wertung mit Fahrernummern zu verbinden, wodurch das Klassement generiert wird.

- **Klassement anzeigen**
  Das durch die eingetragene Wertung erstellte Klassement muss vom *RadioTour Speaker* abgerufen werden können. Dort sollen alle Fahrer angezeigt werden, welche einen Punkterang in diesem Spezialklassement erreichten.

- **Virtuelles Klassement**
  Dem *RadioTour Speaker* muss es möglich sein, ein aktuelles Klassement der Tour abzurufen und dieses nach bestimmten Kriterien zu sortieren. Die zurzeit möglichen Sortierkriterien sind:

  - Gruppen (zur Zeit des Aufrufs, nicht offiziell)
  - Virtueller Rückstand (zur Zeit des Aufrufs, nicht offiziell)
  - Zeitboni (zur Zeit des Aufrufs, nicht offiziell)
  - Offizielle Zeit (zum Etappenende des Vortages, offiziell)
  - Offizieller Rückstand (zum Etappenende des Vortages, offiziell)

Abbildung 7.5: Virtuelles Klassement in der bisherigen Applikation

Ren Spe **Vir** Fah Adm Log **Etappe 3** **Distanz: 0 km**

Rang	Startnr	Fahrername	Team	Land	Gruppen	virtuell Rückst.	Zeit-boni.	offiziell		
								Zeit		Rückst.
1	51	Damiano CUNEGO	Lampre - ISD	ITA		31:01:49 [3]		31:01:49	[1]	31:01:49
2	73	Steven KRUIJSWIJK	Rabobank Cycling Team	NED		31:03:25 [4]		31:03:25	[2]	31:03:25
3	1	Frank SCHLECK	Team Leopard-Trek	LUX		31:03:30 [5]		31:03:30	[3]	31:03:30
4	31	Levi LEIPHEIMER	Team RadioShack	USA		31:03:48 [6]		31:03:48	[4]	31:03:48

- **Fahrerliste anschauen**
  Dem *RadioTour Speaker* muss es möglich sein, die aktuelle Fahrerliste anzuschauen. Die Fahrerliste ist nach Startnummern aufsteigend sortiert. (Die Startnummern werden in Mehretappenrennen so vergeben, dass die Fahrer eines Teams aufeinanderfolgende Startnummern erhalten.)

Abbildung 7.6: Die Fahrerliste mit den Informationen zum Status der Fahrer

Ren Spe Vir **Fah** Adm Log **Etappe 3** **Distanz: 0 km**

Startnummer	Fahrername	Team	Land
1	Frank SCHLECK	Team Leopard-Trek (LEO)	LUX
2	Fabian CANCELLARA	Team Leopard-Trek (LEO)	SUI
3	Jakob FUGLSANG	Team Leopard-Trek (LEO)	DEN

- **Fahrer de- bzw. aktivieren**
  Dem *RadioTour Speaker* muss es möglich sein, einzelne Fahrer zu deaktivieren bzw. wieder zu aktivieren. Der Grund der Deaktivierung soll auch später noch nachvollziehbar sein. Es soll möglich sein, den Grund für die Deaktivierung anzugeben (z.B. ausgeschieden, nicht gestartet, andere). Eine so vorgenommene Deaktivierung eines Fahrers muss durch den *RadioTour Speaker* rückgängig gemacht werden können.

- **Fahrerdetails bearbeiten**
  Dem *RadioTour Speaker* muss es möglich sein, einen Fahrer aus der Fahrerliste auszuwählen, um seine Details anzuschauen und auch zu bearbeiten.

- **Statistik**
  Dem *RadioTour Speaker* muss es möglich sein, in seinem Admin-Bereich eine kurze und prägnante textbasierte Statistik zu erhalten, aus welcher er auf einen Blick sieht, wie viele Fahrer in der Datenbank sind und welche davon aktiv sind. Darüber hinaus die Anzahl Gruppen, Spezialklassemente, Wertungen und vergebene Punkte.

- **Import Fahrerliste**
  Nach jedem Renntag (= Etappe), wird in die *RadioTour* Applikation eine neue Fahrerliste mit den aktuellen offiziellen Zeiten importiert. Deshalb muss es dem *RadioTour Speaker* möglich sein, die aktuellen Zeiten zu importieren. Derzeit sind für den Import der Daten verschiedene Importverfahren implementiert, wie dies im Screenshot ersichtlich ist.

- **Erfassung Spezialklassemente**
  Dem *RadioTour Speaker* muss es möglich sein, Spezialklassemente zu erfassen. Diese Erfassungen werden vor der Tour de Suisse getätigt, weshalb ein Import hier nicht notwendig ist.

- **Speicherung Maillots**
  Dem *RadioTour Speaker* muss es möglich sein, die Belegung der 4 verschiedenen Maillots anzugeben und zu sichern. Folgende Maillots werden an der Tour de Suisse vergeben:

Name	Farbe
Bergpreis	Rot-Weiss
Gesamtklassement	Gelb
Neo-Profi	Weiss
Punkte	Grün

- **Daten exportieren**
  Dem *RadioTour Speaker* muss es möglich sein, die Spezialklassemente, Wertungen, Fahrer nicht nur zu erfassen, importieren und bearbeiten, sondern auch als CSV zu exportieren. Dabei werden einfach alle mit dem gewünschten Export assoziierten Infos in das exportierte CSV geschrieben.

7.5 Usability Test

Mit David Loosli, dem *RadioTour Speaker* an der Berner Rundfahrt 2012, wurde am 11.05.2012 ein Testlauf mit dem Tablet durchgeführt. Dabei wurde an einem vorgängigen Treffen folgende fiktive Situation durchgespielt, um den Umgang mit der Applikation zu erklären.

- Alle Fahrer sind importiert und das Rennen beginnt jetzt
- Rennzeit wird gestartet
- Fahrer 4 & 17 von Beginn an, an der Spitze

- Bereits 1:07 Vorsprung
- 31 hat ein defektes Rad und muss raus
- Der Vorsprung erhöht sich auf 2:19
- 8, 83 & 34 fallen hinter das Feld mit einem Rückstand von 4:31
- 8 kann wieder etwas aufholen und löst sich vom Feld, Rückstand nur noch 3:45
- 7 & 11 sind der Spitze an den Fersen mit einem Rückstand von 0:42
- 13 rückt zu 7&11 auf
- 41 ist verletzt und muss zum Arzt
- Chrono1 ist beim Ortseingang Lyss, ... Jetzt. Stoppuhr
- 37 49 45 rücken in die Verfolgergruppe vor
- Wir sind beim Ortseingang Lyss, Stoppuhr
- 19 & 71 fallen eine Gruppe zurück
- 11 kommt an die Spitze
- Rennzeit wird korrigiert auf 05:11
- 71 kann nicht mehr und hat aufgegeben
- Rückstand vom Feld erhöht sich um 20 s
- Info kommt rein, dass die 3 (Marcel Aregger), eigentlich Michael Aregger heisst
- Aregger (3) kann zur die Spitze aufschliessen
- Hinter dem Feld bildet sich eine neue Gruppe mit den Fahrern 19, 24,
- 51 & 36. Rückstand 3:11
- 19 stürzt
- 24 Rückt wieder ins Feld auf
- 51 und 36 fallen weiter in die nächste Gruppe zurück
- 52 rückt zur Spitze vor
- Rückstand Feld neu 1:30

7.5.1 Auswertung und Feedback

Alle Änderungen konnten durch David Loosli erfasst werden. Der erste Eindruck war sehr gut, folgende Vorschläge wurden aufgenommen und in der Weiterentwicklung umgesetzt:

- Beim TimePicker sind die Nummern zu klein dargestellt, es benötigt keine Stunden

- Fahrer werden nur bei „Aufgabe“ und „nicht erschienen“ ausgegraut und durchgestrichen, bei „Arzt“, „Sturz“ und „Defekt“ werden die Fahrernummern eingefärbt
- ein Fahrer kann die folgenden Status haben
  - im Rennen / aktiv
  - nicht gestartet
  - ausgeschieden
- Tablet darf nicht verdunkeln, falls es einen Moment nicht verwendet wird (Hardwareeinstellung)
- Rennkilometer muss editierbar sein. Weiter sind die noch zu fahrenden Kilometer anzuzeigen
- Rennzeit und Kilometer sind nach einem Absturz der Applikation noch verfügbar

7.6 Kontaktdaten

Dieser Abschnitt enthält die E-Mail-Adressen der an der Arbeit beteiligten Personen. Über diese Adressen können die Personen auch nach der Zeit an der HSR erreicht werden.

Florian Bentele, Student
florian@bentele.me

Daniel Stucki, Student
daniel@stucki.me

Prof. Dr. Peter Heinzmann, Betreuer
peter.heinzmann@cnlab.ch

Lukas Frey, Industriepartner cnlab AG
lukas.frey@cnlab.ch

7.7 Inhaltsverzeichnis der beigelegten CD

Dokumentation zur Studienarbeit RadioTour.pdf

01_Code

- RadioTour.zip
- RadioTourTest.zip
- gippingen_startliste_junit.csv

02_Sitzungsprotokolle

- 20120222_Protokoll-01_SA_RadioTour_V1.1.docx
- 20120305_Protokoll-02_SA_RadioTour_V0.2.docx
- 20120313_Protokoll-03_SA_RadioTour_V0.2.docx

- 20120319_Protokoll-04_SA_RadioTour_V0.1.docx
- 20120402_Protokoll-05_SA_RadioTour_V0.1.docx
- 20120413_Protokoll-06_SA_RadioTour_V0.1.docx
- 20120501_Protokoll-07_SA_RadioTour_V0.1.docx
- 20120510_Protokoll-08_SA_RadioTour_V0.1.docx
- 20120521_Protokoll-09_SA_RadioTour_V0.1.docx
- 20120529_Protokoll-10_SA_RadioTour_V0.1.docx

03_Projektmanagement

- zeiterfassung_radiotour.xls

04_Mockups

- RadioTour_V1.4.pdf

05_Dokumente_pdf

- Aufgabenstellung_RadioTour_V1.0.pdf
- Kaufempfehlung_Android_Tablet.pdf
- BigPicture_mitText.pdf
- Kurzfassung_RadioTour_V1.1.pdf
- ClosePicture_v1.1.pdf
- Poster_RadioTour.pdf
- Developer's Guide RadioTour App.pdf
- Usability_Test.pdf
- Evaluation_iOS_Android.pdf
- Zeiterfassung_Radiotour.pdf
- JSON.pdf

06_Dokumente_org

- BigPicture.docx
- Kurzfassung_RadioTour_V1.1.doc
- BigPicture.vsd
- Poster_RadioTour_V1.1.ppt
- ClosePicture_v1.1.vsd
- Usability Test.docx
- Developer's Guide RadioTour App.docx

- evaluation_iOS_Android.docx
- JSON.docx
- evaluation_iOS_Android.xlsx
- Kaufempfehlung_Android_Tablet.docx
- zeiterfassung_radiotour.xls

07_Dokumente_cnlab

- Beispiel-FahrzeugkolonneTdS.pdf
- Beispiel-Karte_Marschtabelle_TdS2010.pdf
- Beispiel_Profil-Marschtabelle_TdS2010.pdf
- JSON-Tourlive.pdf
- SIM-Card-GPS-SIM-GPRS.pdf
- Technischer_Guide_Mail.pdf

08_Emails

- AWPrototyp.eml
- AWRadioTour.eml
- AWSA Kickoff.eml
- AWSIM-KartefürGPRSTracker.eml
- AWStartlisteBRFSpezialwertungen.eml
- AWTourspeakerBernerRundfahrt.eml
- Bachelorarbeit.eml
- Feedbacks.eml
- HSRSA-Server1.eml
- HSRSA-Server.eml
- KurzfassungundPlakat.eml
- RadioTourBernerRundfahrt - Startliste.eml
- RadioTourBernerRundfahrt.eml
- RadioTour.eml
- Samsung.eml
- TourLive.eml
- WGGippingen2012 1.eml
- WGGippingen2012.eml
- WGStartlisteBRFSpezialwertungen.eml
- WGTourspeakerBernerRundfahrt.eml

7.8 Poster

Für diese Arbeit wurde ebenfalls ein Poster zusammengestellt. Auf der beiliegenden CD befindet sich dieses Poster im pdf Format.

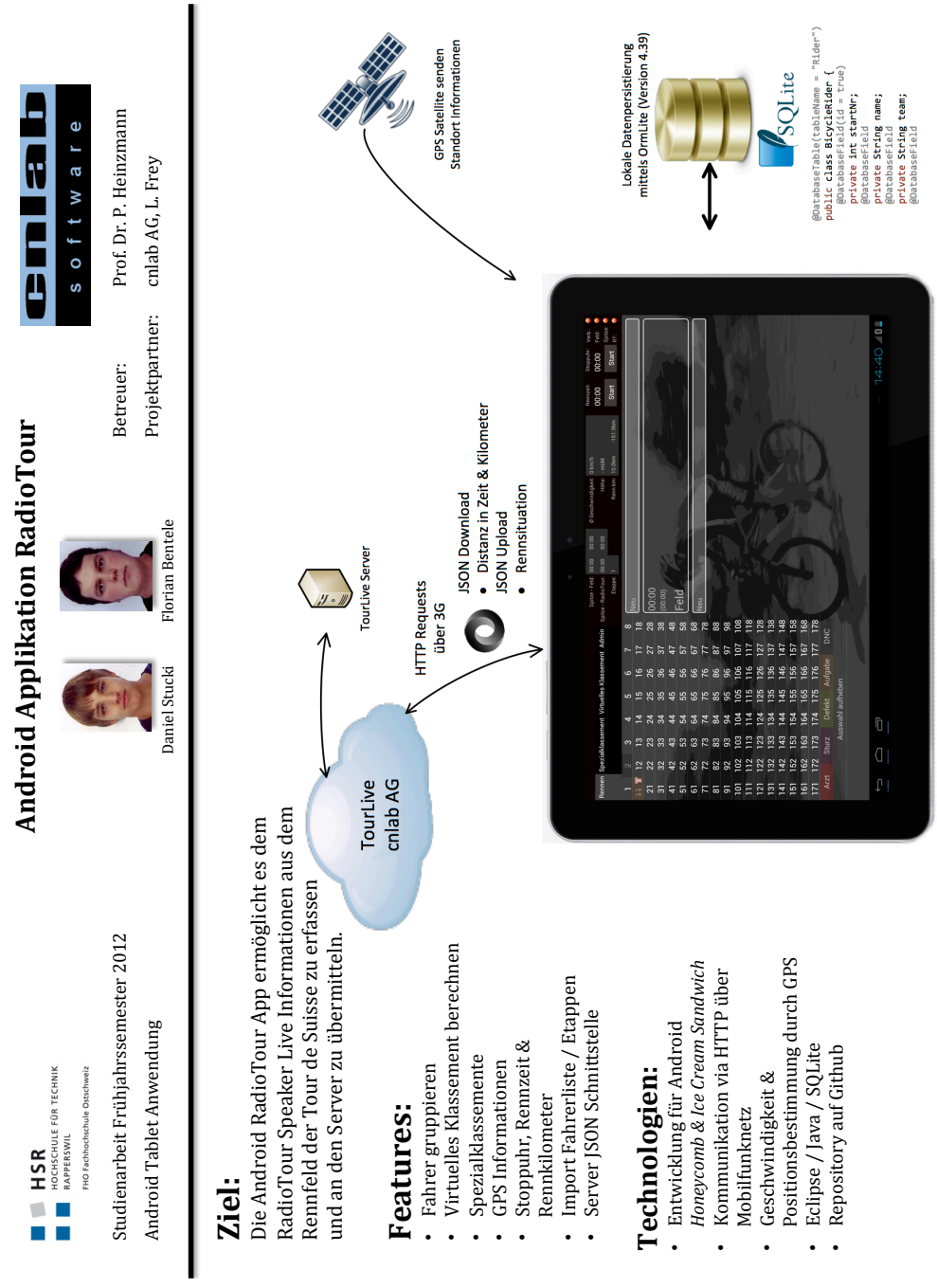


Abbildung 7.7: Poster zum Projekt RadioTour