Machbarkeitsstudie Cyclenator

Kuanlun Huang, Christian Maran, Thomas Pokorny, Markus Schulmeister

# Versionsverzeichnis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Author | Datum | Status | Kommentar |
| 0.1 | Markus Schulmeister | 11.10.2012 | Neu |  |
| 0.2 | Markus Schulmeister | 8.11.2012 | Erweitert | Produktbeschreibung, Ist-Zustand, Soll-Zustand, Funktionsumfang hinzugefügt |
| 0.3 | Christian Maran | 14.11.2012 | Erweitert | Inhaltsverzeichnis, Wirtschaftliche Machbarkeit, Nutzenanalyse |
| 0.4 | Thomas Pokorny | 15.11.2012 | Erweitert | Technische Machbarkeit |
| 1.0 | Markus Schulmeister | 15.11.2012 | Erstversion fertig | Conclusio |
| 1.0.1 | Markus Schulmeister | 22.11.2012 | Erweitert | Deckblatt |
| 1.1 | Markus Schulmeister | 17.01.2013 | Überarbeitet | Soll-Zustand komplett überarbeitet; googleMaps zur technischen Machbarkeit hinzugefügt; Funktionsumfang überarbeitet, sinnfreie Grafik durch genaue Beschreibungen ersetzt; Conclusio aktualisiert |

# Inhaltsverzeichnis

1. **Produktbeschreibung**
2. **Ist-Zustand**
3. **Sollzustand**

3.1 Muss-Ziele

3.2 Kann-Ziele

1. **Funktionsumfang**
   1. Must Have
2. **Technische Machbarkeit**
   1. Technologien
3. **Wirtschaftliche Machbarkeit**
   1. Personalaufwand
      1. Qualifikation der Teammitglieder
      2. Zusammenarbeit des Projektteams
      3. Personelle Risiken
      4. Projektaufwand pro Person
   2. Investitionsaufwand
4. **Nutzenanalyse**
   1. Nutzen für den Kunden
   2. Nutzen für das Projekt-Team
5. **Conclusio**

# 1 Produktbeschreibung

Das Programm soll die optimale Lösung für alle Outdoor begeisterten Sportler wer-den. Einerseits korrigiert die App die Schwächen der bereits vorhandenen Produkte und fügt gleichzeitig neue Funktionalitäten hinzu.

Benutzer können mittels der App eine eigene Rad- bzw. Laufstrecke erstellen, oder diese von dem App suchen lassen und diese ihren Wünschen entsprechend optimieren. Dazu gehört das Hinzufügen von Streckenpunkten, das Auswählen des Schwierigkeitsgrades, ob die Strecke als Runde oder Hin- und Rückfahrt absolviert wird und vieles mehr.

Angeboten wird die App im Google Play-Market wo sie heruntergeladen werden kann.

Angewendet wird das Programm auf allen Android-Tablets und Smartphones. Die Zielgruppe besteht, wie bereits erwähnt, vor allem aus Hobbysportlern.

# 2 IST-Zustand

Gängige Mobileroutenplaner wie Runtastic, OpenCycle und Outdoor haben Vorzüge, jedoch viele Mängel, vor allem was die Optimierung für den Benutzer bezüglich der Strecken angeht(z.B.: Die Länge der Strecke, Entfernung zur Strecke, Anzahl der Wegpunkte, Rad oder Laufstrecke)

# 3 SOLL-Zustand

## 3.1 Muss-Ziele

**Erweiterbarkeit**

Cyclenator soll möglichst erweiterbar aufgebaut werden. Die Grundanwendung selbst hat lediglich die Aufgabe die Karten und die aktuelle Position anzuzeigen. Die weiteren Funktionen wie Streckenplanung etc. werden in zusätzlichen Klassen programmiert. Auf diese Art und Weise bleibt das Projekt erweiterbar um z.B. nächstes Jahr als Projekt fortgesetzt zu werden.

**Kompatibilität ab Android-Version 2.1**

Die App soll auf allen Geräten einwandfrei funktionieren, auf welchen zumindest die Android-Version 2.1 läuft. Version 2.1 ist die bisher am meisten verbreitete Version von Android und neue Geräte kommen mit höheren Versionen. Daher haben wir uns für 2.1 entschieden, da es den besten Kompromiss zwischen Verbreitung und Aktualität bildet.

## 3.2 Kann-Ziele

**Mehrsprachigkeit**

Die App wird erst mal auf Deutsch umgesetzt, da es für uns einfacher ist. Wenn die App kommerziell vermarktet werden soll, muss sie allerdings auch auf Englisch angeboten werden. Deswegen haben wir uns entschlossen die Ausgabetexte dynamisch zu laden um die Sprache der App später einfach erweitern zu können.

# 4 Funktionsumfang

## 4.1 Must have

**/LF10/** Strecken nach bestimmten Kriterien suchen

z.B.: Länge der Strecke, Entfernung zur Strecke, Anzahl der Wegpunkte, Rad oder Laufstrecke

Diese Funktion ist die einzige Funktion die explizit vom Auftraggeber verlangt wurde, daher wird ihr Nutzen als besonders hoch eingeschätzt. Der Aufwand wird als gering eingeschätzt, da wir Erfahrung mit Datenbankabfragen haben.

**/LF20/** Gesamtlänge der Strecke berechnen und anzeigen

Vor dem Beginn der Tour wird dem Benutzer die Gesamtlänge der Route angezeigt, zur Laufzeit wird dynamisch die noch zu absolvierende Strecke angezeigt.

Die Gesamtlänge wird einfach durch addieren der Abstände zwischen den Waypoints berechnet und zur Laufzeit werden die Abstände zwischen den verbleibenden Waypoints und der Abstand zum nächsten Waypoint addiert. Der Abstand zum nächsten Waypoint wird wie in /LF70/ beschrieben berechnet. Diese Funktion wird nicht als besonders aufwändig eingeschätzt, da es sich Großteils um simple Addition handelt.

**/LF30/** Strecken speichern

Der User soll die Möglichkeit haben häufig benutzte Strecken zu speichern und diese später wieder aufzurufen. Um eine Strecke zu speichern müssen alle Waypoints zur zugehörigen Strecke in einer Datenbank gespeichert werden

Der Aufwand dieser Funktion liegt eher im Schreibaufwand als der Komplexität, da wir alle Erfahrung mit Datenbanken haben und wird daher nicht als besonders hoch eingeschätzt.

**/LF40/** Zwischen Runden und Hin- und Rückfahrt entscheiden

Strecken sollen als Runden oder Hin- und Rückfahrt absolviert werden können.

Die Unterscheidung zwischen Runde und Hin- und Rückfahrt macht programmiertechnisch eigentlich nur den Unterschied das die meisten Waypoints doppelt verwendet werden und somit weniger Waypoints gespeichert werden müssen. Der Aufwand hierfür ist nicht besonders hoch und der Nutzen ist eine bessere Performance.

**/LF50/** Strecke auf Waypoints aufteilen

Eine Strecke besteht aus Waypoints, die nacheinander abgefahren werden. Der Benutzer soll die Möglichkeit haben selbst Strecken aus Waypoints zu erstellen oder eine vorgeschlagene Strecke um Waypoints zu erweitern oder diese zu löschen.

Die Grundfunktion hierfür wird von der Google Maps API zur Verfügung gestellt und muss dann noch auf die App zugeschnitten werden, was wir als mittel-aufwändig einschätzen.

**/LF60/** Seehöhe mit Höhenunterschied und Steigung zum nächsten Waypoint anzeigen

Die Seehöhe bekommen wir von der Google Maps API, womit sich der Höhenunterschied und die Steigung dann berechnen lässt, also keine sehr aufwendige Funktion.

**/LF70/** Strecke bis zum nächsten Waypoint oder Ende der Route anzeigen

Dem User werden der noch verbleibende Weg und die Richtung bis zum nächsten Waypoint angezeigt.

Für den noch verbleibenden Weg wird der Abstand zwischen der aktuellen Position und dem nächsten Waypoint berechnet. Die Richtung wird über die Längen- und Breitenkoordinaten mithilfe des Kompasses berechnet und angezeigt. Das schätzen wir als relativ aufwendig ein.

**/LF80/** Joule- oder Kalorienverbrauchs und Durchschnittsgeschwindigkeit anzeigen

Nach dem die Strecke abgefahren wurde wird dem Benutzer sein Joule- bzw. Kalorienverbrauch und die Durchschnittsgeschwindigkeit angezeigt.

Die Durchschnittsgeschwindigkeit kann einfach über die Gesamtdistanz der Strecke und der benötigten Zeit berechnet werden. Der Kalorienverbrauch mittels Dauer, Steigung und Durchschnittsgeschwindigkeit. Das sind simple Berechnungen, also nicht sehr aufwändig zu programmieren.

# 5 Technische Machbarkeit

## 5.1 Technologien

### OpenCycleMap

OpenCycleMap ist eine globale Karte für Fahrradfahrer, basiert auf Daten vom OpenStreetMap-Projekt. Es wurde in 2007 gestartet und seitdem immer mehr erweitert.

Weder Registrierung noch API-key ist notwendig um die Karten zu benutzen.

Kommerzielle Nutzung ist gestattet und sogar erwünscht.

OpenCycleMap stellt eine ausführliche Developer Beschreibung auf ihrer Website zur Verfügung

### Outdooractive

Zum Touren planen.

Stellt seine Java-Script API zur Verfügung.

Um die API nutzen zu dürfen muss man seine Anwendung bei outdooractive registrieren.

Man verpflichtet sich auch einen Link auf deren Website in die Anwendung und auf die Website der Anwendung einzubauen.

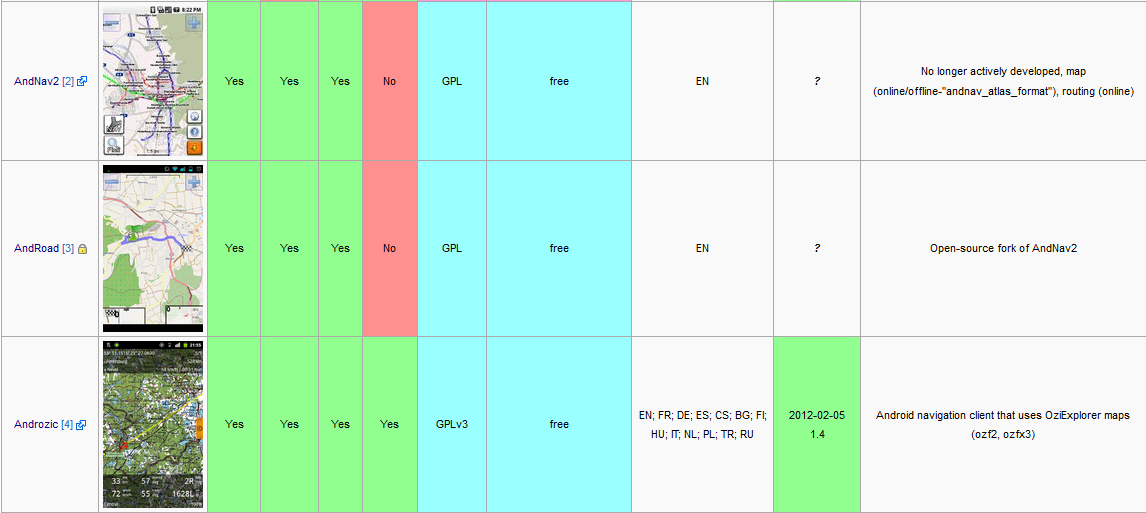
Outdooractive bietet weiters die Möglichkeit auf Wunsch kundenspezifische Karten zu erstellen.

### OpenSteetMap

OpenStreetMap ist ein freies Projekt, das für jeden frei nutzbare Geodaten sammelt (Open Data). Mit Hilfe dieser Daten können Weltkarten errechnet oder Spezialkarten abgeleitet werden, sowie Navigation betrieben werden. Auf der OpenStreetMap-Startseite ist eine solche Karte abrufbar.

Der Kern des Projekts ist eine wiki-ähnliche Datenbank mit geographischen Daten. Diese dürfen gemäß der Open Database License verwendet werden. Dadurch ist eine Einbindung in Drucke, Webseiten und Anwendungen wie Navigationssoftware möglich, ohne durch restriktive Lizenzen beschränkt zu sein oder Entgelte zahlen zu müssen. Die Nennung von OpenStreetMap als Datenquelle ist zur Datennutzung erforderlich.

Mittels OpenStreetMap wurden bereits zahlreiche Android Apps programmiert, die meisten von ihnen sind Opensource. Liste von Referenz Programmen:



### Android SDK

Programmiert wird das App für alle Android Versionen ab 2.1, das entspricht der der SDK Version 7. Die API stellt alle wichtigen Grundfunktionen die zur Aufbau der APP benötigt werden zur Verfügung. Alle Projektmitglieder haben die SDK bereits installiert und schon Erfahrung im Android-JAVA Bereich gesammelt.

### Google Maps Android API

Mit der Google Maps Android API kann man Karten, basiert auf Google Maps Daten, zu seiner Applikation hinzufügen. Die API kümmert sich automatisch um den Zugriff zu den Google Maps Servern, Datendownload, Kartendarstellung und Reaktion der Karte zu Touchscreen-Input. Man kann die API auch dazu benutzen Markierungen, Polygone etc. zu einer Karte hinzuzufügen.

Applikationen, die die Google Maps API benutzen, dürfen kostenlos bis zu 25000 map loads pro Tag generieren. Als map load zählt:

* Eine Karte, welches die Maps JavaScript API (V2 or V3) benutzt, wird von einer Webseite oder Applikation dargestellt.
* Ein Street View panorama, welches die Maps JavaScript API (V2 or V3) benutzt, wird von einer Webseite oder Applikation, welche nicht auch eine Karte anzeigt, dargestellt.
* Eine SWF, die die Maps API for Flash ladet, wird von einer Webseite oder Applikation geladen.
* Eine einzelne Anfrage für ein Kartenbild von der Static Maps API.
* Eine einzelne Anfrage für ein Panoramabild von der Street View Image API.

Überschreitungen dürfen nicht länger als 90 Tage andauern.

# 6 Wirtschaftliche Machbarkeit

## 6.1 Personalaufwand

### 6.1.1 *Qualifikation der* Teammitglieder

Da die Projektmitglieder über das nötige technische Know-How verfügen und maximal bei bestimmten Sachen ihr Wissen auffrischen müssen, stellt es kein Problem dar das Produkt zu entwickeln. Diese Kenntnisse wurden in der bisherigen Ausbildung sowohl theoretisch als auch praktisch anhand diverser Projekte und Praktika bei Firmen erworben.

### 6.1.2 Zusammenarbeit des Projektteams

Alle Teammitglieder waren sich von Anfang an bewusst, dass nur durch eine starke Zusammenarbeit die von unserem Projektauftraggeber und vom Team gesetzten Ziele erreicht werden können.

Die teaminterne Zusammenarbeit spielt eine besonders wichtige Rolle für eine gerechte und gezielte Aufteilung der Arbeitspakete. Des Weiteren hilft eine teamintern gute Absprache zur Vermeidung bzw. Lösung von auftretenden Problemen bei.

### 6.1.3 Personelle Risiken

#### Ausfall eines Teammitglieds

Das Gelingen eines Projektes kann auch vom Ausfall eines oder mehrerer Teammitglieder abhängig sein. Das kann viele Gründe haben, sei es durch Krankheiten oder andere soziale Faktoren. Die Risiken liegen hierbei vor allem bei der zeitgerechten Abwicklung des Projekts und auch beim möglichen Verlust von fachspezifischen Wissensquellen.

#### Streit im Team

Streit im Team ist heutzutage keine Seltenheit. Es ist schwer ihn zu vermeiden, da es eine Menge Faktoren gibt, die einen Streit innerhalb eines Teams auslösen können wie etwa Meinungsverschiedenheiten, soziale Spannungen usw., die im schlimmsten Fall auch zur Auflösung des Teams führen können. Hierbei sind die Fähigkeiten des Projektleiters gefragt, der derartige Probleme frühzeitig erkennen und diese dann lösen muss.

### 6.1.4 Projektaufwand pro Person

Zur Bestimmung des Projektaufwands pro Person verwenden wir die Funktion-Point-Analyse.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Kategorie | Anzahl | Klassifizierung | Gewichtung | Zeilensumme |
| Eingabedaten | 12 | Einfach | 3 | 36 |
|  | Mittel | 4 | 0 |
|  | Komplex | 6 | 0 |
| Abfragen | 5 | Einfach | 3 | 15 |
|  | Mittel | 4 | 0 |
| 1 | Komplex | 6 | 6 |
| Ausgaben | 8 | Einfach | 4 | 32 |
|  | Mittel | 5 | 0 |
| 2 | Komplex | 7 | 14 |
| Datenbestände | 2 | Einfach | 7 | 14 |
| 1 | Mittel | 10 | 10 |
|  | Komplex | 15 | 0 |
| Referenzdaten | 2 | Einfach | 5 | 10 |
| 1 | Mittel | 7 | 7 |
|  | Komplex | 10 | 0 |
| Summe |  |  | E1= | 144 |
| Einflussfaktoren  (ändern den Function-Point-Wert um ± 30% | 1.Verflechtung mit anderen Anwendungssystemen (0-5) | | | 0 |
| 2.Dezentrale Daten und dezentrale Verarbeitung (0-5) | | | 4 |
| 3.Transaktionsrate (0-5) | | | 0 |
| 4. Verarbeitungslogik  a Rechenoperationen (0-10)  b Kontrollverfahren (0-5)  c Ausnahmeregelungen (0-10)  d Logik (0-5) | | |  |
| 8 |
| 0 |
| 0 |
| 1 |
| 6. Wiederverwertbarkeit (0-5) | | | 5 |
| 6. Datenbestandskonvertierungen (0-5) | | | 0 |
| 7.Anpassbarkeit (0-5) | | | 3 |
| Summe der 7 Einlüsse | E2= | | | 21 |
| Faktor EInflussbwertung | E3=E2/100+0,7= | | | 0,91 |
| Bewerte Function-Points | E1\*E3= | | | 131,04 |
| Aufwand pro Person | E4=Personalmonate/Anzahl der Teammitglieder | | | 2,75 |

Nachfolgend ist eine Zuordnungstabelle zwischen den bewerteten Function-Points und den daraus abgeleiteten Personalmonaten zu sehen.

|  |  |
| --- | --- |
| Bewertete Function-Points | Personalmonate |
| 50 | 5 |
| 100 | 8 |
| 150 | 11 |
| 200 | 14 |
| 250 | 17 |
| 300 | 20 |
| 350 | 24 |
| 400 | 28 |
| 450 | 32 |
| 500 | 36 |

Wenn man nun die Werte vergleicht kommt man in unserem Projekt auf einem Gesamtaufwand von 11 Monaten und wenn man das dann noch durch die Anzahl der Teammitglieder teilt auf 2,75 Monate.

## 6.2 Investitionsaufwand

Der Investitionsaufwand ist relativ gering da nur Vermarktungskosten im Google Play Store und das Personalgehalt aufkommen.

|  |  |
| --- | --- |
| Personal | |
| Personalgehalt | 15x480€ |
| Summe | 7200€ |
| Google Play Store | |
| Vermarktungskosten | 20€ |
| Summe | 20€ |
| Gesamtkosten | 7220€ |

# 7 Nutzenanalyse

## 7.1 Nutzen für den Kunden

* Eine speziell für Hobbysportler entwickelte Android-App zur Radroutenplanung
* Persönliches Feedback für den Benutzer durch Abruf der vollbrachten Leistungen
* Leichte Routenplanung und Wiederverwendbarkeit alter Routen

## 7.2 Nutzen für das Projekt-Team

* Erweiterung der fachlichen Programmierkenntnisse mittels eines praxisnahen Software-Projekts
* Wiederverwendbarkeit von Teilen der entwickelten Software

# 8 Conclusio

Es wird empfohlen alle Must have-Funktionen mittels Java und der Google Maps API in Eclipse mit der Android SDK umzusetzen.